NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SIMULATION STUDY OF TRAFFIC FLOW AT A THREE WAY INTERSECTION

by

Chong Chul Song

September 1988

Thesis Advisor: Peter A. W. Lewis

Approved for public release; distribution is unlimited

Unclassified

security classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification Unclassified	1b Restrictive Markings
2a Security Classification Authority	3 Distribution Availability of Report Approved for public release; distribution is unlimited.
2b Declassification Downgrading Schedule	
4 Performing Organization Report Number(s)	5 Monitoring Organization Report Number(s)
6a Name of Performing Organization Naval Postgraduate School — 6b Office Symbol *(if applicable)* 52	7a Name of Monitoring Organization Naval Postgraduate School
6c Address *(city, state, and ZIP code)* Monterey, CA 93943-5000	7b Address *(city, state, and ZIP code)* Monterey, CA 93943-5000
8a Name of Funding Sponsoring Organization — 8b Office Symbol *(if applicable)*	9 Procurement Instrument Identification Number
8c Address *(city, state, and ZIP code)*	10 Source of Funding Numbers Program Element No — Project No — Task No — Work Unit Accession No

11 Title *(include security classification)* SIMULATION STUDY OF TRAFFIC FLOW AT A THREE WAY INTERSECTION

12 Personal Author(s) Chong Chul Song

13a Type of Report	13b Time Covered	14 Date of Report *(year, month, day)*	15 Page Count
Master's Thesis	From To	September 1988	105

16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms *(continue on reverse if necessary and identify by block number)*
Field	Group	Subgroup	Intersection,arrival,service,queue,simulation.
		[illegible]ized	

19 Abstract *(continue on reverse if necessary and identify by block number)*

This study develops simulation models for the evaluation of the traffic performance at a three way intersection with no signal. In particular, the models were designed to test and to evaluate the wartime road capacity of the current and a proposed road system in the Korean theater. The models describe how traffic at the road intersection will vary with changes to the input variables and priority rules. An analysis is performed to compare performance measures of the intersection given various traffic conditions. Specifically, the analysis will enable an efficient modification of the roads or an appropriate road usage plan in wartime.

[illegible]: Ground traffic; Arrival; Queuing theory; [illegible] Routing; [illegible] programs; Service [illegible] (Theses. edc)

20 Distribution Availability of Abstract ☒ unclassified unlimited ☐ same as report ☐ DTIC users	21 Abstract Security Classification Unclassified
22a Name of Responsible Individual Peter A. W. Lewis	22b Telephone *(include Area code)* (408) 625-1665 — 22c Office Symbol 55Lw

DD FORM 1473.84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete — security classification of this page

Unclassified

Approved for public release; distribution is unlimited.

Simulation Study of Traffic Flow at a Three Way Intersection

by

Chong Chul Song
Major, Republic Of Korea Army
B.S., Korea Military Academy, 1980

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1988

Author: ______________________
Chong Chul Song

Approved by: ______________________
Peter A. W. Lewis, Thesis Advisor

____
Keebom Kang, Second Reader

____
Peter Purdue, Chairman,
Department of Operations Research

____
Kneale T. Marshall,
Dean of Information and Policy Sciences

ABSTRACT

This study develops simulation models for the evaluation of the traffic performance at a three way intersection with no signal. In particular, the models were designed to test and to evaluate the wartime road capacity of the current and a proposed road system in the Korean theater. The models describe how traffic at the road intersection will vary with changes to the input variables and priority rules. An analysis is performed to compare performance measures of the intersection given various traffic conditions. Specifically, the analysis will enable an efficient modification of the roads or an appropriate road usage plan in wartime.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

A. BACKGROUND

The wartime traffic on the roads in the southern theater of the Republic of Korea needs special attention, especially the traffic involved in a war which is very dynamic, and in particular a war which involves a sudden attack from the north and the necessity of falling back to the south. There are several main avenues leading from the front, from which many streets to the east or west are derived, so we have many three way intersections with undeveloped roads. The traffic on these roads is not very heavy in peace time, but it appears as if the limited road capacity may restrict the timely movements of forces in a time of emergency.

The study of traffic in this specific area has never been done before because under the current strategy no forces are allowed to withdraw, with the exception of supply forces and forces needing to be replaced. We assume that in many cases all the forces will have to move simultaneously or in a certain sequence. For instance, the enemy may select battlefields which are quite different from those we expect. Also, the deployed forces may move forward or backward simultaneously as the situation changes. It is strongly believed that traffic congestion will occur in the assumed contingencies, and therefore, it is necessary to have some idea of the dynamics involved. A particular question which could be asked is, under what condition will the traffic moving onto the main road from the side road be unstable. That is, when will the queue of traffic continue to grow in an unstable manner. This thesis presents a very limited approach to the problem of the dynamics of force withdrawal from the front or a deployment of reserve forces to the front, but it is an important first step to the understanding and solution of the whole problem of traffic control and flow in the Korean theater of operations.

B. DESCRIPTION OF THE PROBLEM

The purpose of building a stochastic simulation model is to describe traffic performance at a three way intersection. This type of intersection is a principal capacity limiter and main source of delay of the current road system in the Korean theater. At first, an analytical model for the current road system under special simplifying assumptions is developed, including a transition diagram and brief solution procedure. We will

assume that a unit, which possibly consists of many vehicles, is treated as a single customer as in general queueing problems. Each unit's transition time is equated with service time in the sense of a classical queueing situation, [Refs. 1,2: pp. 13-20]. The interarrival and service times of a unit are assumed to have certain respective distributions, which will be discussed in detail later. Further, the interaction of different units at the intersection will be controlled by traffic policies to be described in Chapter II.

Due to difficulties encountered in the analytical approach, two simulation models are developed for the current and a proposed road system. We are concerned with finding how the traffic performance at the intersection varies with changes to the arrival and service time distributions. We consider performance in terms of the average waiting time of the units and the average number of units waiting in each direction of the intersection.

The intersection system must be thoroughly understood because blockages at an intersection may cause serious problems in conducting operations in an area where units are deployed with high density relative to the road capacity. The units moving or waiting because of blockages are very vulnurable to enemy ambushes since they are not fully ready for engagements.

C. SCOPE OF THE THESIS

In attempting to achieve the goals of this study, an analytical model is developed up to the transition diagram, together with the general solution procedure outlined in Chapter II. It is found that the simulation model is needed to solve the problems because of the difficulties in handling various inputs and because of the limited solution capability available for the analytical model.

Two simulation models are built with SIMSCRIPT II.5 in order to conduct the simulation experiments. The first one is based on the current road system and the other is for a proposed modification to the system. The two models are presented in Chapter III and Chapter IV. Figure 1 and Figure 15 depict the current and proposed road systems respectively, with possible movement routes.

Chapter V will analyze and compare the results of the two simulation models. The analysis is done by exporting output files of SIMSCRIPT runs to the GRAFSTAT program on the IBM 3033AP main frame. This allows one to graphically examine the models for stability of the performance measurements and to examine the results in the

stable cases as a function of several input factors. Finally, conclusions and recommendations are presented in Chapter VI.

II. PROBLEM STATEMENT

As mentioned before, we can apply this model to the wartime traffic in a theater of operations where the road capacity is limited in comparison with the potential road users. We assume that there are no traffic signals and no military police for traffic control and assume services are controlled by service policies we set up for the simulation study. It is the fundamental road capacity and traffic performance at the intersection that we want to examine, therefore, we must fully understand the following assumptions to properly grasp the specific features and logic of the model. A current intersection is presented in Figure 1.

A. ASSUMPTIONS

1. Units

A unit consists of a group of vehicles, the number of which depends on the organization of forces and the tactics of a war. All the vehicles in the unit will be treated as one customer at the intersection (general queueing facility) because the movement of a combat unit such as a company or a battery must be conducted under tight command and control to minimize confusion and loss of vehicles. Thus we assume that no unit or vehicle can intervene between another unit's vehicles.

2. Arrivals and Queues

Poisson arrivals are assumed for the analytical model. Each arrival from the three directions is filed in an appropriate queue if it is not its turn for service. Our intersection queueing facility has three queues; a north, south and east queue and there is no interaction between vehicles in each of the queues, that is, a First Come First Served principle is effective within each direction. Actually the time of arrival is that of the unit to the end of the line, potentially some distance away from the intersection. Since units move up to the intersection in close order as units which arrived previously are served, the distance and stop and start effects of the queue at the intersection are ignored. In fact, the waiting time of a unit is proportional to its distance from the intersection. This could be exploited in the analysis but will be ignored.

3. Route Selection

Possible routes for a unit from each direction are shown by arrows in Figure 1. One of two routes will be selected for a unit from each direction and this selection is made when a unit becomes the first unit in the queue and no unit is presently served

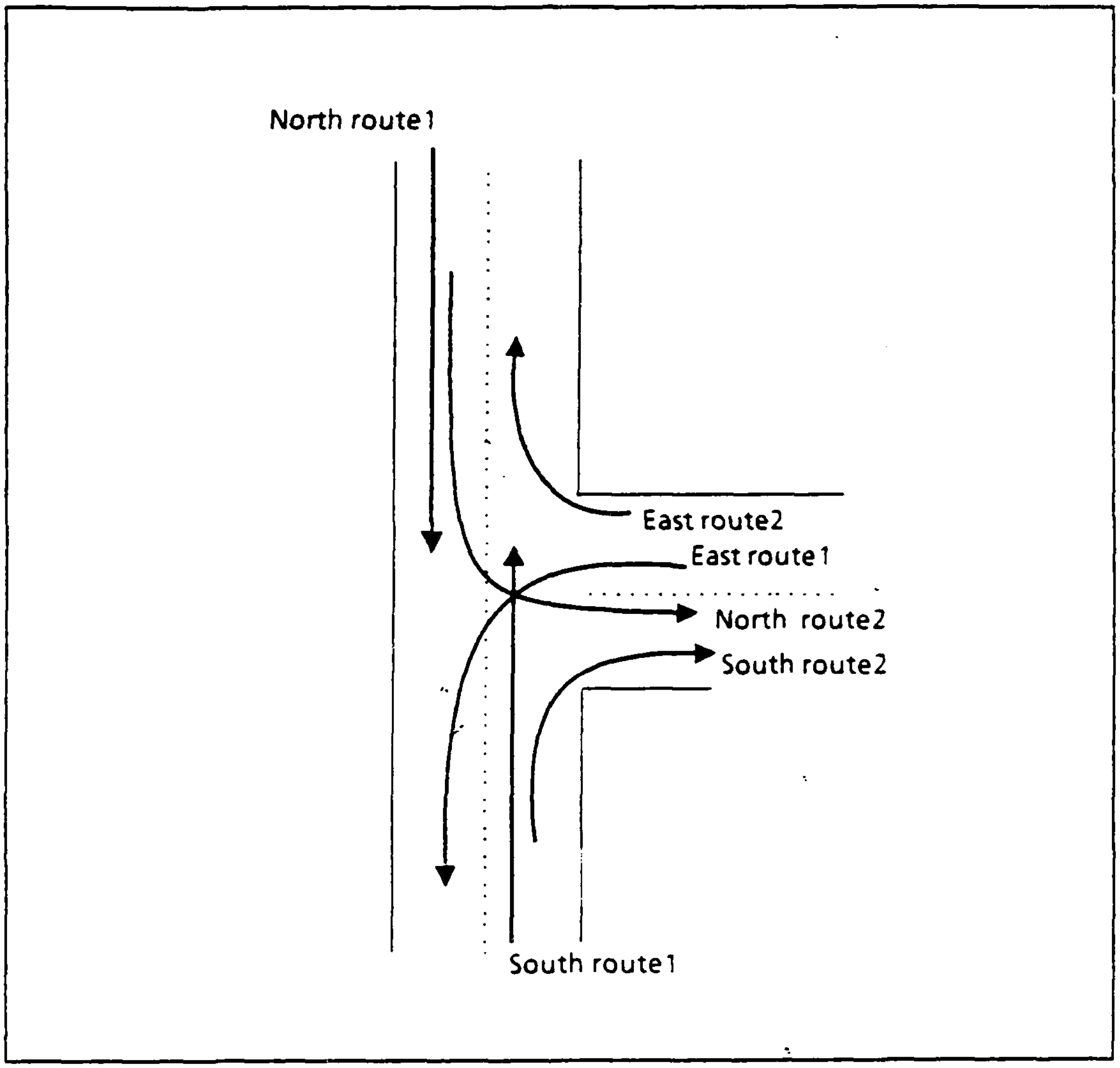

Figure 1. **A current Intersection.:** Traffic coming from the north either continues south or turns east if it is not blocked. Similarly traffic from the south can continue north or turn east if it is not blocked and traffic from the east turns either north or south. Priorities for service on the routes will be discussed in detail in Chapter II.

from the direction or when an arriving unit sees no unit in the queue. The probability used to decide a route is an input variable in the model and a uniform (0,1) random variable U will be drawn to determine the route selection. If U is less than the probability P of taking route1, route1 is selected, otherwise route2 is selected. The probabilities

of taking north route1, south route1, and east route1 may be different from each other and are user input parameters.

4. Service

Service begins when the first vehicle of a unit starts from the intersection and ends upon the passage of the last vehicle. Once served a unit is assumed to leave the system, even though, in reality, it may significantly affect the service conditions of follow on units. Service times depend on many factors, such as the length of a unit, road conditions, weather, number of refugees and so on. Though many of the factors should be studied separately, they are assumed to be compounded into a single random variable used for service time in this model. Our main interest in this thesis is the interactions between units at the intersection. We assume the service times for the six routes are independent and exponentially distributed in the analytical model. This is done for analytical tractability and is probably not physically valid. More general and probably more realistic assumptions are made in the simulation models.

5. Service Policies for Model 1

In Model 1, service for a unit starts immediately upon arrival when there is no blocking to the selected route, otherwise it is filed in the queue. There are two possible routes from each direction named route1 and route2 as shown by arrows in Figure 1. Suppose that the set (north route, south route, east route) indicates the state of movements at the intersection where each entity within that set can have value 0, 1 or 2 according to the routes being used. Here 0 indicates that neither of the two possible routes from each direction are used; otherwise 1 and 2 represent the route number used. It is clear that all combinations of 3^3 (= 27) states are not included in the possible movement states of the model because of the interactions between the routes. For instance, the state (1,1,2) never happens since south route1 and east route2 can not be used simultaneously.

In Model 1, in fact there are only 14 states of movement which are acceptable: (0,0,0), (0,0,1), (0,0,2), (0,1,0), (0,2,0), (0,2,1), (0,2,2), (1,0,0), (1,0,2), (1,1,0), (1,2,0) (1,2,2), (2,0,0), and (2,0,2). Even though these sets are not themselves enough to study the quantities of interest, the service polices are based on these possible movement combinations.

A movement on a certain route may block more than one route to which units have to move, therefore the priorities for movement must be specified according to the route for which a service is just completed. Note that we have assumed there is no traffic signals and no military police for traffic control. A unit desiring to move in the system must wait until its route is clear and it has priority (in the case of ties). We break ties in cases in which more than one unit has conflicting priorities simultaneously. The routes that block movement and the next priority for a service given a blocking unit which has completed service are shown in Table 1.

Table 1. PRIORITIES AT THE END OF SERVICE OF A BLOCKING ROUTE (MODEL 1)

Route which is about to finish service	Next Priority, in order, at the end of blocking
north route1	east route1, (north route1,2)
north route2	east route1, (south route1,2), (north route1,2)
south route1	(east route1,2), north route2, (south route1,2)
south route2	north route2, (south route1,2)
east route1	(north route1,2), south route1, (east route1,2)
east route2	south route1, (east route1,2)

The left column of each row in Table 1 indicates the routes on which service was completed and the right column of each row identifies all possible routes which can be selected by units waiting to pass when service was completed by the route denoted in the left column. Priorities for next service are given to the right column routes in the order listed. We set up the specific priorities for this model by breaking ties and placing the routes for which service was just completed last in order of priority. Next priorities at the end of service on each route will be discussed in detail below. The routes in the parenthesis denote the two possible candidates from the direction blocked, so only one of the two routes can be selected for the first unit in the queue. Table 1 also shows that a newly arriving unit desiring to move on a route of a left column and row can only

move if all the right column routes of the same row are not used, otherwise it has to wait for service until next priority is given to it.

After service for a unit on a route in the left column of Table 1 is completed, the priority for next service is conditioned on other movements at the intersection. As an example, the third row could represent units that have to move to (east route1 or east route2), north route2, and (south route1 or south route2) and are blocked by a unit presently using the left column south route1. Upon service completion on south route1 the first check is to determine if there is blocking of east route1 or east route2. Let's assume that a unit from the east has been waiting for movement to the north (east route2 in the first position of the right column), the unit can move immediately because east route2 is always clear right after the service completion of south route1. Next, service for north route2 (second position in order) will be considered and a unit waiting to go from the north to the east (north route2) can also move immediately because there is no blocking to that route at the end of south route1's service regardless of the service on east route2. Note that east route2, which has the first priority for service, does not block north route2. Finally, service on the south routes (third position in priority order) will be considered and movement to either south route1 or south route2 will be conditioned on the status of east route2 and north route2. If we had assumed east route1 (the other route in the first parenthesis of the right column in the third row) was waiting to move instead of east route2, we must check to determine if there is a unit on north route1 upon the service completion of south route1. If north route1 is active this route also blocks east route1. If there is no unit on north route1, east route1 can be used, otherwise service on the next route in the order of priority, north route2, must be considered. In other words, even though a unit has priority upon a route's service completion, the unit loses priority if it is blocked by route presently in service.

In Table 2 through Table 7 we tabulate cases of the priority problems. The triple in the row designator of each Table describes the routes selected by units which are specifically blocked by the route that is about to finish service. The first element refers to the traffic from the north, the second refers to traffic from the south and the third refers to traffic from the east. In each case the element can have value 0, 1 or 2, where 0 indicates no traffic waiting from that direction, 1 indicates traffic waiting for route1 and 2 indicates that traffic is waiting for route2. The triples in the column designator represent all possible movement states when the route of interest, which is specified in the Table title, is about to finish service. Finally, N1, N2, S1, S2, E1, and

E2 represent north route1,2, south route 1,2, and east route 1,2 respectively. A route identified at the intersection of a row and column has priority upon the service completion of the specified unit in the title of the Table. N/A indicates that the combination of the row and column condition is not feasible in this Table. A 0 at the intersection of a row and column indicates that no additional route can be used immediately after service completion of the unit specified in the Table title.

Table 2. SERVICE PRIORITIES AT THE END OF A NORTH UNIT MOVING SOUTH (denoted by north route1)

N S E	1 0 0	1 0 2	1 1 0	1 2 0	1 2 2
0 0 1	E1	N/A	0	E1	N/A
1 0 0	N1	N1	N1	N1	N1
1 0 1	E1	N/A	N1	E1	N/A
2 0 0	N2	N2	0	0	0
2 0 1	E1	N/A	0	E1	N/A

In the last row of Table 2 the entry (2 0 1) in the left column means that when service on north route1 is completed, north route2 is selected for the next north unit (the first 2), there is no one waiting from the south direction (the second 0) and someone is waiting to go from the east to the south (the last 1). The priorities for movement of waiting units must be conditioned on the present situation at the intersection, and this is the column indicator. Thus while a unit from the north is moving south, a unit from the south can be moving north (south route1, designated by the second 1 in the header of column 3) if no unit from the east is turning north (east route2), and can move east at any time (south route2). Therefore there are only five movement states possible when north route1 is about to finish service. The priorities for next service given the row entry are dependent on which one of the five movement states describes the intersection just prior to the completion of north route1. Not all row and column combinations are feasible and are designated by N/A. Thus in row (2 0 1) and column (1 0 2), east route2 is being used so no route is selected for an east unit (the 1 of (2 0 1) can't be selected). A 0 indicates that no additional route can be used after service completion of north route1. Thus for row (2 0 0) and column (1 2 2), the unit from the north can not move east because a unit from the south is moving east (2 in the second position of the column heading).

Table 3. **SERVICE PRIORITIES AT THE END OF A NORTH UNIT TURNING EAST** (denoted by north route2): Explanation of Table arrangement is provided on pages 8 and 9.

N S E	2 0 0	2 0 2
0 0 1	E1	N/A
0 1 0	S1	0^{∇}
0 1 1	E1	N/A
0 2 0	S2	S2
0 2 1	S2 and E1	N/A
1 0 0	N1	N1
1 0 1	E1	N/A
1 1 0	N1 and S1	$N1^{\nabla}$
1 1 1	E1	N/A
1 2 0	N1 and S2	N1 and S2
1 2 1	E1 and S2	N/A
2 0 0	N2	N2
2 0 1	E1	N/A
2 1 0	S1	$N2^{\nabla\nabla}$
2 1 1	E1	N/A
2 2 0	S2	S2
2 2 1	E1 and S2	N/A

While a unit from the north is turning east (north route2), only east route2 is not blocked by the unit. Thus we have only two possible movement combinations at the intersection just prior to north route2 finishing service. All combinations of the 6 routes except east route2 are in the row designators, which indicate the possible routes selected by units waiting to pass when service on north route2 is completed. A unit on east route2 (the last 2 in column (2 0 2)) still blocks a unit waiting for service on south route1 when north route2's service is completed, thus south route1 loses priority, which is designated by ∇ . The case of $\nabla\nabla$ in row (2 1 0) and column (2 0 2) indicates that the priority initially given to south route1 is transferred to north route2 because of the unit on east route2.

Table 4. SERVICE PRIORITIES AT THE END OF A SOUTH UNIT MOVING NORTH (denoted by south route1): Explanation of Table arrangement is provided on pages 8 and 9.

N S E	0 1 0	1 1 0
0 0 1	E1	0^{∇}
0 0 2	E2	E2
0 1 0	S1	S1
0 1 1	E1	$S1^{\nabla}$
0 1 2	E2	E2
0 2 0	S2	S2
0 2 1	S2 and E1	$S2^{\nabla}$
0 2 2	S2 and E2	S2 and E2
2 0 0	N2	N/A
2 0 1	E1	N/A
2 0 2	N2 and E2	N/A
2 1 0	N2	N/A
2 1 1	E1	N/A
2 1 2	N2 and E2	N/A
2 2 0	N2	N/A
2 2 1	E1 and S2	N/A
2 2 2	N2 and E2	N/A

South route1 blocks all routes except north route1. Thus we have two possible movement combinations just prior to south route1 finishing service and all combinations of routes which have been blocked by a unit on south route1 in the row designators to identify priorities at the end of service on south route1. A unit from the north moving south (the first 1 of column (1 1 0)) blocks east route1 again after service completion on south route1. This is designated by ∇ in this Table. As an example, in row (0 2 1) and column (1 1 0), priority initially given to east route1 must be transferred to south route2 because north route1 is being used and blocks east route1 when service on south route1 is completed.

Table 5. SERVICE PRIORITIES AT THE END OF A SOUTH UNIT TURNING EAST (denoted by south route2)

N S E	0 2 0	0 2 1	0 2 2	1 2 0	1 2 2
0 1 0	S1	0	0	S1	0
0 2 0	S2	S2	S2	S2	S2
2 0 0	N2	$0^{\nabla\nabla}$	N2	N/A	N/A
2 1 0	N2	$0^{\nabla\nabla}$	N2	N/A	N/A
2 2 0	N2	$S2^{\nabla}$	N2	N/A	N/A

Explanation of Table arrangement is provided on pages 8 and 9. A unit on south route2 blocks only a unit from north turning east (north route2) and units from the south routes themselves. Therefore north route2 has the priority for next service if there is no unit moving on east route1, else the priority will again be transferred to the south routes. This is designated by ∇ in row (2 2 0) and column (0 2 1). Note that in Table 1, north route2 is placed ahead of the south routes in order of priority at the end of service on south route2. This is because we have assumed that once service is completed in one direction, priority for next service will be given to the other directions which have been blocked by the unit leaving the system. A unit on the east routes, the last 1 or 2 of the triples in columns 2, 3, and 5 allows no service on south route1 and the movement on east route1 (1 of column (0 2 1)) at the end of south route2's service also blocks north route2 (see $\nabla\nabla$). The 0's in rows 1, 3, and 4 and columns 2, 3, and 5 are because units on east route1,2 continue to block service of units attempting to move at the completion of south route2's service.

Table 6. SERVICE PRIORITIES AT THE END OF AN EAST UNIT TURNING SOUTH (denoted by east route1): Explanation of Table arrangement is provided on pages 8 and 9.

N S E	0 0 1	0 2 1
0 0 1	E1	E1
0 0 2	E2	E2
0 1 0	S1	N/A
0 1 1	S1	N/A
0 1 2	S1	N/A
1 0 0	N1	N1
1 0 1	N1	N1
1 0 2	N1 and E2	N1 and E2
1 1 0	N1 and S1	N/A
1 1 1	N1 and S1	N/A
1 1 2	N1 and S1	N/A
2 0 0	N2	0
2 0 1	N2	$E1^{\nabla}$
2 0 2	N2 and E2	$E2^{\nabla}$
2 1 0	N2	N/A
2 1 1	N2	N/A
2 1 2	N2 and E2	N/A

Only a unit from the south moving east (south route2) is allowed service while a unit from the east is moving south (east route1). Therefore we only have to consider two movement combinations just prior to east route1 finishing service and all possible combinations of units waiting for east route1's service completion. One or two units may receive priority when service on east route1 is completed according to the service policies discussed earlier. The service on south route2, the second 2 of (0 2 1) in column 2, blocks a unit from the north waiting to move east (north route2). Thus the priority initially given to north route2 must be transferred to the east routes, which is designated by ∇ in this Table.

Table 7. SERVICE PRIORITIES AT THE END OF AN EAST UNIT TURNING NORTH (denoted by east route2)

N S E	0 0 2	0 2 2	1 0 2	1 2 2	2 0 2
0 0 1	E1	E1	0	0	0^{∇}
0 0 2	E2	E2	E2	E2	E2
0 1 0	S1	N/A	S1	N/A	0^{∇}
0 1 1	S1	N/A	S1	N/A	0^{∇}
0 1 2	S1	N/A	S1	N/A	$E2^{\nabla}$

Explanation of Table arrangement is provided on pages 8 and 9. A unit in service on east route2 blocks only south route1 and east route1,2. Therefore south route1 has the priority for next service if there is no unit in service on north route2, else the priority will be again transferred to the east routes. Note that in Table 1, south route1 places ahead of the east routes in order of priority at the end of service on south route2 because we have assumed that priority for next service is given to the other directions once service from one direction is completed. The unit on north route2, the first 2 of column (2 0 2) blocks a unit waiting for service on east route1 and south route1, which is designated by ∇.

B. ANALYTICAL MODEL

1. System Notations

In this section, we introduce the notations required to develop an analytical model. Note that subscripts of n, s, and e represent north, south and east respectively.

north units	Units from the north direction.
λ_n	Poisson arrival rate at the intersection from the north direction.
μ_n	Exponential service rate of a unit from the north direction.
$Q_n(t)$	Number of north units waiting for service, as a function of time t.
Next.north.route	Next route selected for the first unit in north.queue.
north.queue	Number of units waiting for service from the north and next route for the first unit in the north queue.
north.run	Route that a north unit uses now.
P_n	Probability that a north unit will take north route1.

Note that north.queue represents two quantities, $Q_n(t)$ and next.north.route with "/" between them. As an example, "3/1" of north.queue indicates that there are 3 units in the queue and the first unit in the queue is supposed to use north route1.

There are also south units, λ_s, μ_s, south.queue, south.run, next.south.route, Q_s and P_s for the south direction and east units, λ_e, μ_e, east.queue, east.run, next.east.route, Q_e and P_e for the east direction.

2. A Transition Diagram

Suppose that potential units arrive in accordance with Poisson rate λ_n, λ_s, λ_e from 3 directions and that service rates are exponential μ_n, μ_s, and μ_e. We can then theoretically compute the quantities of interest for the system; the mean number of units in the queues and the average amount of time that an arriving unit waits in the queue. To accomplish this we must first decide upon an appropriate state space. It is clear that the state of the system must include more information than merely the number of units in each queue of the system and the present movement state (1 of 14 movement states we discussed earlier). For instance, to decide the movements after a service completion, it would not be enough to specify that there are n_1, n_2, n_3 units in each queue and which routes are being used, but we would also have to know the routes selected for the first unit in the queues.

To account for this, we define a state variable (north.queue, south.queue, east.queue, north.run, south.run, east.run), where the first three entities give information about the number of units in the queues and the desired routes for the first units in the

queues and the last 3 entities, north.run, south.run and east.run represent the routes that are presently being used by north, south, or east units. For example, the following sample states have the following interpretation.

(0/0, 2/1, 1/0, 0, 0, 2) :

The entry 0/0 means that there is no unit in the north queue.

The entry 2/1 means that there are two units in the south queue and south route1 is selected for the first unit of south queue.

The entry 1/0 means that there is one unit in the east queue. A route is not decided for the unit because an east unit is still being served.

The last three entries 0, 0, 2 means that only east route2 out of the six routes is presently being used. Note it blocks south route1.

(0/0, 2/1, 1/0, 1, 0, 2) :

The conditions are the same as the above state except a new north unit uses north route1 (the 0 in position four has changed to a one).

(0/0, 1/0, 1/2, 1, 1, 0) :

The entry 1/0 means that there is one unit in the south queue. A route is not decided for the unit because a south unit is still being served.

The entry 1/2 means that there is a unit in the east queue and east route2 is selected for the unit.

The entry 1 in position four means that a north unit uses north route1.

The entry 1 in position five means that a south unit uses south route1.

It should be noted that the process will go to the second state (0/0, 2/1, 1/0, 1, 0, 2) from the first state (0/0, 2/1, 1/0, 0, 0, 2) when a unit arrives at the intersection from the north and takes north route1 (which occurs at rate $P_n \lambda_n$). Also the process will go to the third state (0/0, 1/0, 1/2, 1, 1, 0) from the second state (0/0, 2/1, 1/0, 1, 0, 2) when the east unit completes its service and east route2 is selected for the following unit (which occurs at rate $(1 - P_e)\mu_e$).

As discussed earlier, there are 14 combinations of routes which are allowed in the system. Anyone of these combinations may have many states since we have approximately $\{3(n + 1)\}^3$ states for each combination of the routes where n is the length of the queues. We can begin to analyze the states from the following development of the (0,0,1) route combination.

(0/0, 0/0, 0/0, 0, 0, 1)
(0/1, 0/1, 0/1, 0, 0, 1)
(0/2, 0/2, 0/2, 0, 0, 1)

(1/0, 1/0, 1/0, 0, 0, 1)
(1/1, 1/1, 1/1, 0, 0, 1)
(1/2, 1/2, 1/2, 0, 0, 1)

.

.

.

(n/0, n/0, n/0, 0, 0, 1)
(n/1, n/1, n/1, 0, 0, 1)
(n/2, n/2, n/2, 0, 0, 1)

Of course in this model all $\{3(n+1)\}^3$ states are not possible in the state space of the model. For instance, the state (1/1, 1/1, 1/1, 0, 0, 1) is not in the state space since an east unit is still being served, and the direction of the first unit in the east queue has not been designated i.e., the third entry must be 1/0 if a unit from the east is being served. As a result, each route combination may have less than $\{3(n+1)\}^3$ states but this is still too many states to describe in a transition diagram for a whole model. In addition, it is not straightforward to build a total state space since we would have to consider all interactions of the routes for each movement state.

Rather than a direct attempt to solve the whole problem we can observe how the transitions occur and what the balance equations look like by starting from the initial point, that is, the base state (0/0, 0/0, 0/0, 0, 0, 0). A brief transition diagram that describes a small part of the whole model is shown in Figure 2.

3. Solution Procedures

Up to now we have decided upon the state space and tried to make a transition diagram, although they are not fully described. Also we have assumed Poisson arrivals and an exponential service rate for the system, so the model can be analyzed as a continuous-time Markov chain, [Ref 3: pp. 210-273].

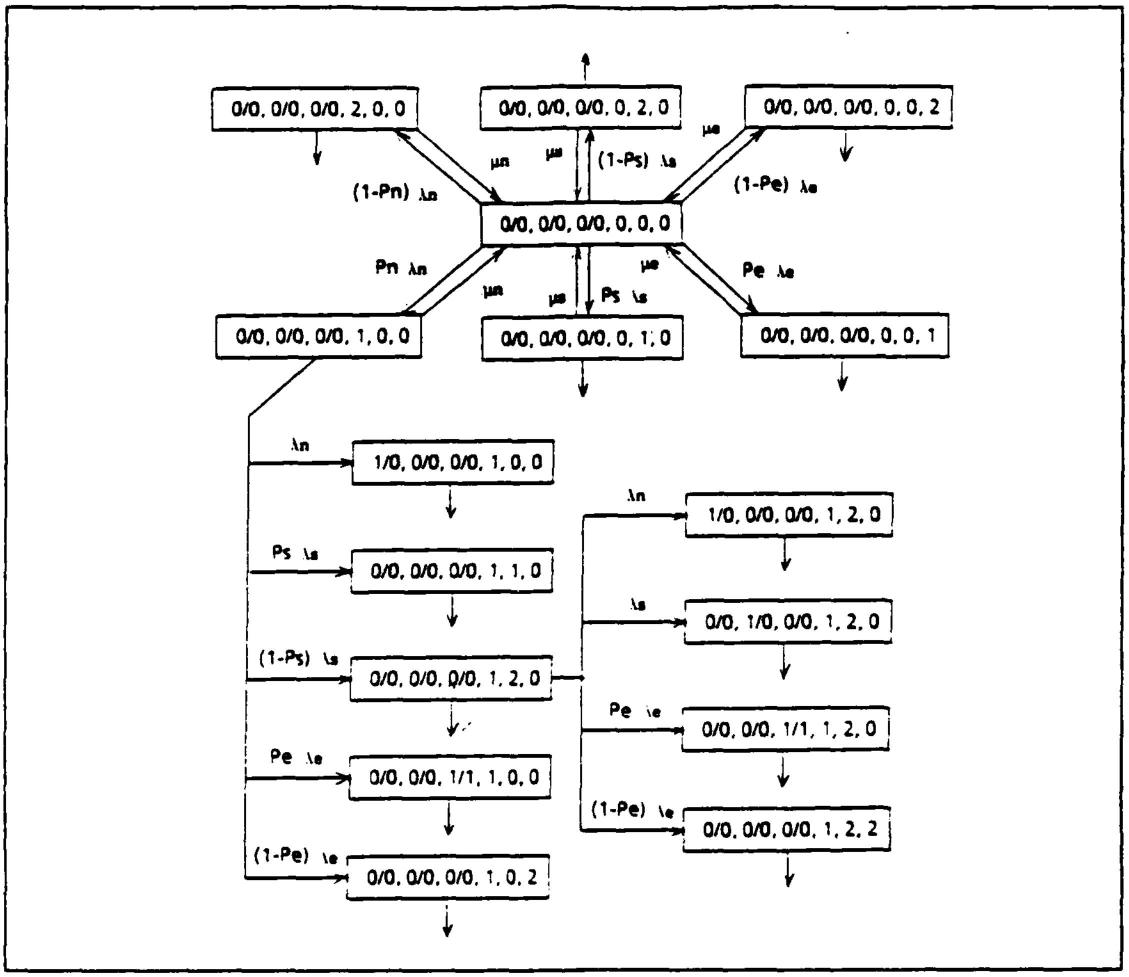

Figure 2. A partial transition diagram of the model: The short arrows at the bottom of a state represent many more arrows pointing in and and out of the state. The notations of Pn, λn, and μn are used for p_n, λ_n, and μ_n. The notations of Ps, λs, and μs are used for the three parameters of the south and Pe, λe, and μe are used for the three parameters of the east direction.

Consider limiting probabilities p_j for all j states. $p_j \equiv \lim_{t \to \infty} p_{ij}(t)$ where j is independent of state i and p_{ij} is the probability that the process enters state j from state i. If we assume that the limiting probabilities p_j exist, then they can be obtained by the following formula, [Ref. 4: pp. 246-248].

$$v_j p_j = \sum_{k \neq j} v_k p_{kj} p_k \quad \text{for all state } j, k. \tag{2.1}$$

$$\sum_j p_{j=1}, \quad \text{for all state } j, \text{ where} \tag{2.2}$$

$v_j p_j$: Rate at which the process leaves state j.

$\sum_{k \neq j} v_k p_{kj} p_k$: Rate at which the process enters state j.

v_j : Leaving rate from state j.

v_k : Leaving rate from state k.

p_j : Proportion of time the process is in state j.

p_k : Proportion of time the process is in state k.

p_{kj} : Probability of transitions from state k to state j.

Once we solve the balance equations, it is easy to answer our questions in terms of these limiting probabilities. However, as we see in Figure 2, the number of states derived from the base state increases geometrically. Suppose the maximum number of "n" is 9, then we have approximately $(10 \times 3)^3 \times 14 = 415{,}800$ states and the same number of balance equations where 14 indicates the number of all possible combinations of movements. It is not straightforward to develop the transition diagram and to write all the balance equations and as a result, the approach to solving the problem necessitates the simulation method.

III. SIMULATION MODEL 1

A. ASSUMPTIONS AND NOTATIONS

The assumptions and notations of simulation Model 1 remain the same as those for the analytical model discussed in Chapter II, except that the interarrival and service times may now have a Gamma distribution with various parameters, [Ref 5: pp. 267-273].

B. INPUTS TO THE MODEL

Our primary goal is to examine the waiting time and the number of units in the queue of the steady state under certain conditions given by specified input parameters. The simulation model is made ready to run by entering the necessary input values. This part of the model is extremely important since it provides the user an opportunity to specify the particular conditions which are to be examined, as well as the values which establish the boundary conditions of the simulation. These input values represent features of the system such as the probability of taking route1 and the shape of the service and interarrival time distributions. Hence, for Model 1, we have a total of three factors which we divide into different levels in Table 8 for the purpose of controlling the size of the simulation.

We assume that successive service and interarrival time distributions for the three directions are independent and identically distributed. In addition, we assume the probability of taking route1 is the same for each direction. Thus from Table 8 we have 2 levels of θ_1, k_1 and k_2, 3 levels of P_1 and 3 levels of traffic intensity. Thus we initially simulated (levels of θ_1) $\times$ (levels of k_1) $\times$ (levels of k_2) $\times$ (levels of P_1) $\times$ (levels of ρ) $=$ $2 \times 2 \times 2 \times 3 \times 3 = 72$ cases of system conditions. Note that we have 4 different combinations of the Gamma shape parameters k_1 and k_2; (1,1), (1,4), (4,1), (4,4). The value 1 for k_1 or k_2 represents the case where service time or interarrival times are skewed, i.e. the arrivals form a Poisson process and the service rate is exponential, as in the analytical model. The value 4 for the shape parameter indicates that the Gamma distribution is regular, [Ref 6: p. VIII.7].

Table 8. MAJOR FACTORS AND LEVELS OF SIMULATION.

Factors	Levels
Service time distributions (denoted by the subscript 1 on parameters θ and k)	Gamma(θ_1,k_1) where θ_1 = 2.5, 5.0 and $k_1 = 1, 4$
Traffic intensities ρ ($\rho = \frac{\text{mean service time}}{\text{mean interarrival time}}$)	$\rho = 0.5, 0.4, 0.33$
Interarrival time distributions (denoted by the subscript 2 on parameters θ and k)	Gamma(θ_2,k_2) where $\theta_2 = \frac{\theta_1}{\rho}$ and $k_2 = 1, 4$
Probabilities of taking route1	P_1 = 0.25, 0.5, 0.75

C. MODEL IMPLEMENTATION

1. Simulation Language

The model is written in Simscript II.5, which has become the standard simulation language for military applications and also a powerful and flexible simulation tool in all areas of military modeling. Simscript II.5 reduces simulation programming time and cost several fold compared to FORTRAN, and is enormously more efficient and flexible than interpretive simulation techniques, [Ref. 7,8].

2. Model Structure

a. General

The program contains a preamble, a main, 4 routines and 9 event routines. The 4 routines include read.data, initialize, single.run.report, and sum.report. The 9 event routines are north(south,east).unit.generator, new.north(south,east).unit, and north(south,east).service. Routines are called by the main routine or by other routines and return control to the calling routine. Event routines are special types of subroutines which are accessed by being scheduled, [Ref. 9,10]. The functions of each routine and interactions among routines are as follows:

b. Preamble

It contains the definitions of entities, attributes and sets, and other global information.

c. Main

This is the first routine to receive control during the execution of a simulation run. Its function is to initialize the system and to combine the input parameters for performing a series of simulations. It also computes and prints some simulation statistics.

d. Read.data

This routine reads all the input parameters provided by a user. A user specifies these parameters in the main routine. The input parameters are three seed stream numbers, the desired number of units to be simulated, the desired number of replications, the various Gamma distribution parameters for the interarrival and service times, and the probability of taking route1 for each direction of the intersection. In this thesis we examine only the cases described in Table 8.

e. Initialize

The spaces for all array elements are allocated. It also schedules initial events.

f. Single.run.report

All information of a single simulation run is collected here and a verification of the model can be done by examining this report. A sample report is shown in Table 11 of Appendix A.

g. Sum.report

This routine summarizes the output from single.run.report. The report can be used to note the stationary statistics and to observe how the output statistics change with respect to the input parameters. In addition it can be used to analyze the boundary conditions for stability; this being one of the research objectives. The study of this report is used to determine an appropriate design size for Model 1. A sample section of the resulting report is shown in Table 12 of Appendix A.

h. North(south,east).unit.generator

Note that south and east in the parenthesis indicate that the same routines also exist for the south and east directions. New.north.unit event routine is scheduled immediately after the north.unit.generator event routine is called and this event itself is rescheduled for a next arrival. The interarrival time for the next north arrival is a

Gamma random variate whose parameters, as described previously, are inputs to the model. The termination of simulation is also controlled by this event routine after the desired number of units have been simulated from each of the three directions.

i. New.north(south,east).unit

A north unit is created immediately after this event routine is called and this unit is either served directly or filed in the queue. Whether it is served or filed in the queue and which route is selected for the unit, depends on other movements at the intersection and the route selection procedures we discussed in Chapter II (see route selection procedure on page 4). If the unit is served immediately, a north.service event routine is scheduled to occur at the end of service. Note that the service time is a Gamma random variate.

j. North(south,east).service

A unit that has passed the intersection is destroyed immediately and this unit leaves the system. The units blocked by the leaving unit are served according to the service priorities discussed in Chapter II and the appropriate north(south,east).service event routines will be scheduled based on new service times (Gamma random variates).

3. Logic of the Model

The flow charts of 4 major event routines for the north and a macro flow chart for the whole model are depicted in Figure 3 through Figure 7 respectively. The flow charts for other directions are omitted to avoid redundancy. The logic of the event routines for the south and east is exactly the same as that for the north, though the conditions for movement to a given route is quite different.

We can divide the north(south,east).service event routine into two phases; a route selection phase for the first unit in the queue and a decision phase for the next movements. There is no difference between the north, south or east directions in the route selection phase. However, the decision phase is quite different for each direction since each route blocks different routes and the conditions of movement to a blocked route is different. As should be apparent at this point, the decision phase of the north(south,east).service event routine must correspond with the service policies and priority rules discussed earlier. The program listing is presented in Appendix B.

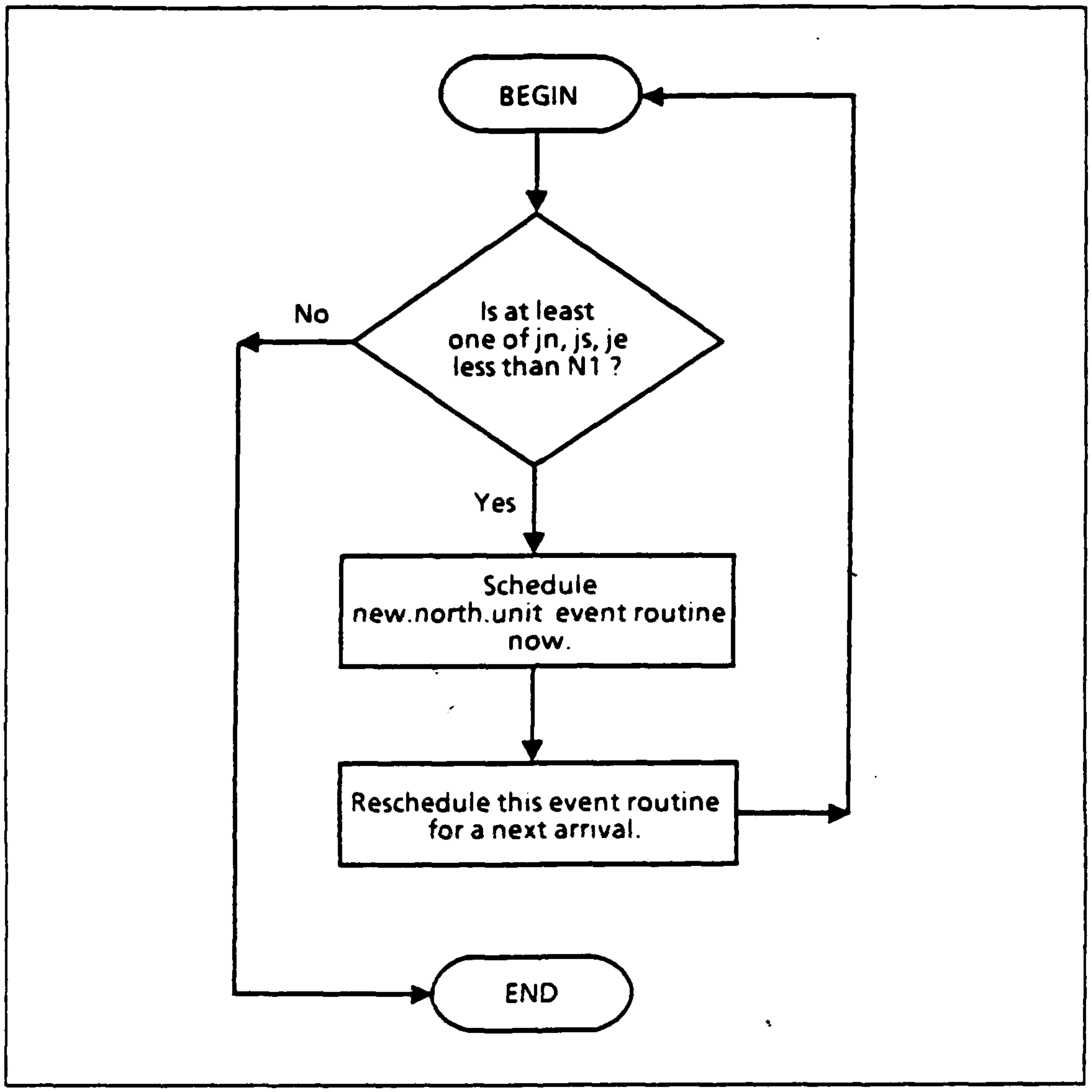

Figure 3. **Flow chart for north.unit.generator event routine:** Jn is the number of north.units that have passed the intersection, js is for south.units and je is for east.units. N1 is the minimum desired number of units to be simulated from each direction and is selected by a user. This routine will be rescheduled as long as at least one of jn, js or je is less than N1 and will stop being rescheduled if all three values are greater than or equal to N1.

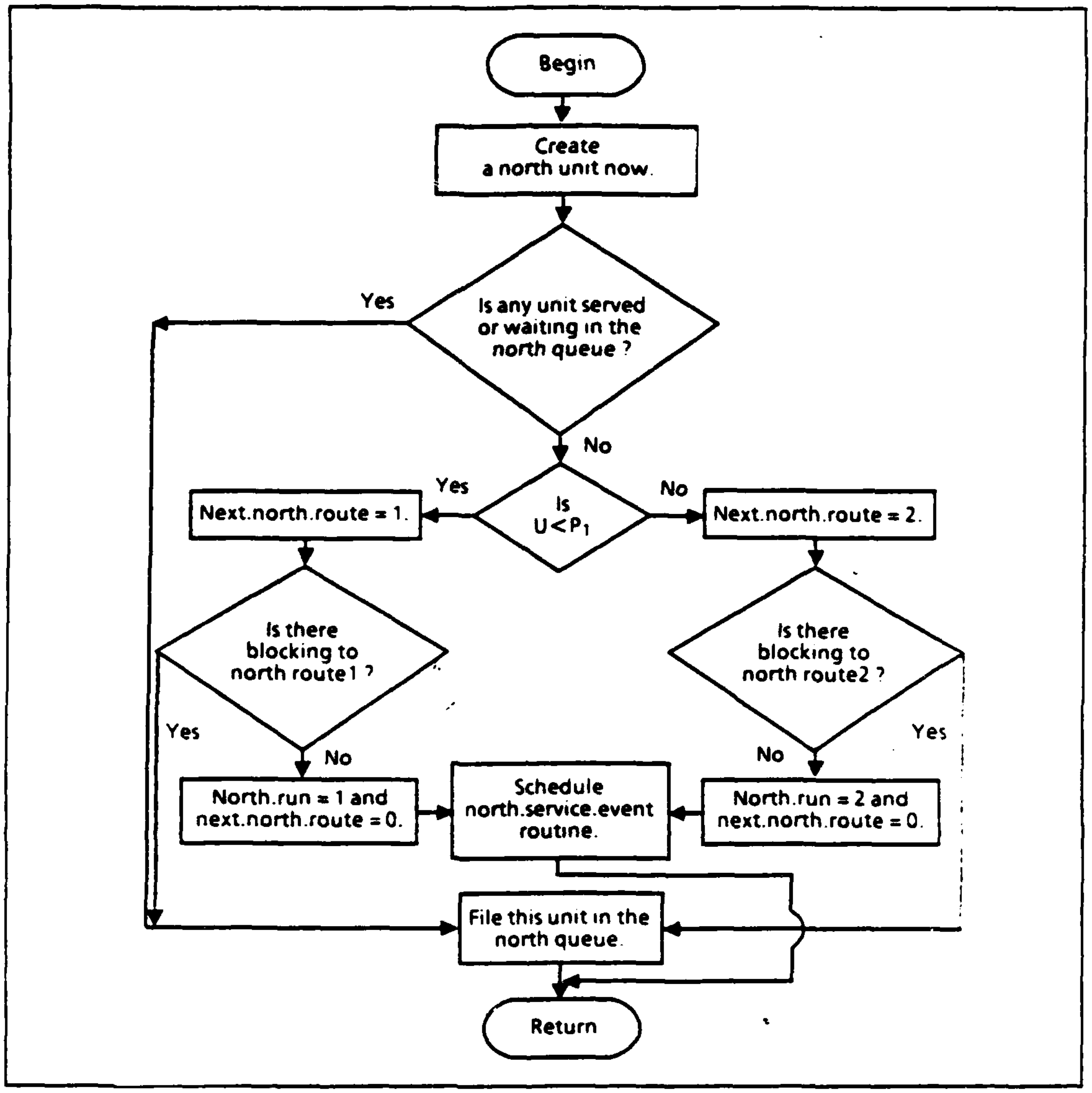

Figure 4. Flow chart for new.north.unit event routine: U is a Uniform (0,1) random variate and is compared with P_1 for route selection of the unit where P_1 is the probability of taking route1. An arriving unit which is created in this routine can be served immediately or filed in the queue depending upon queue condition, route selection and the other movements at the intersection.

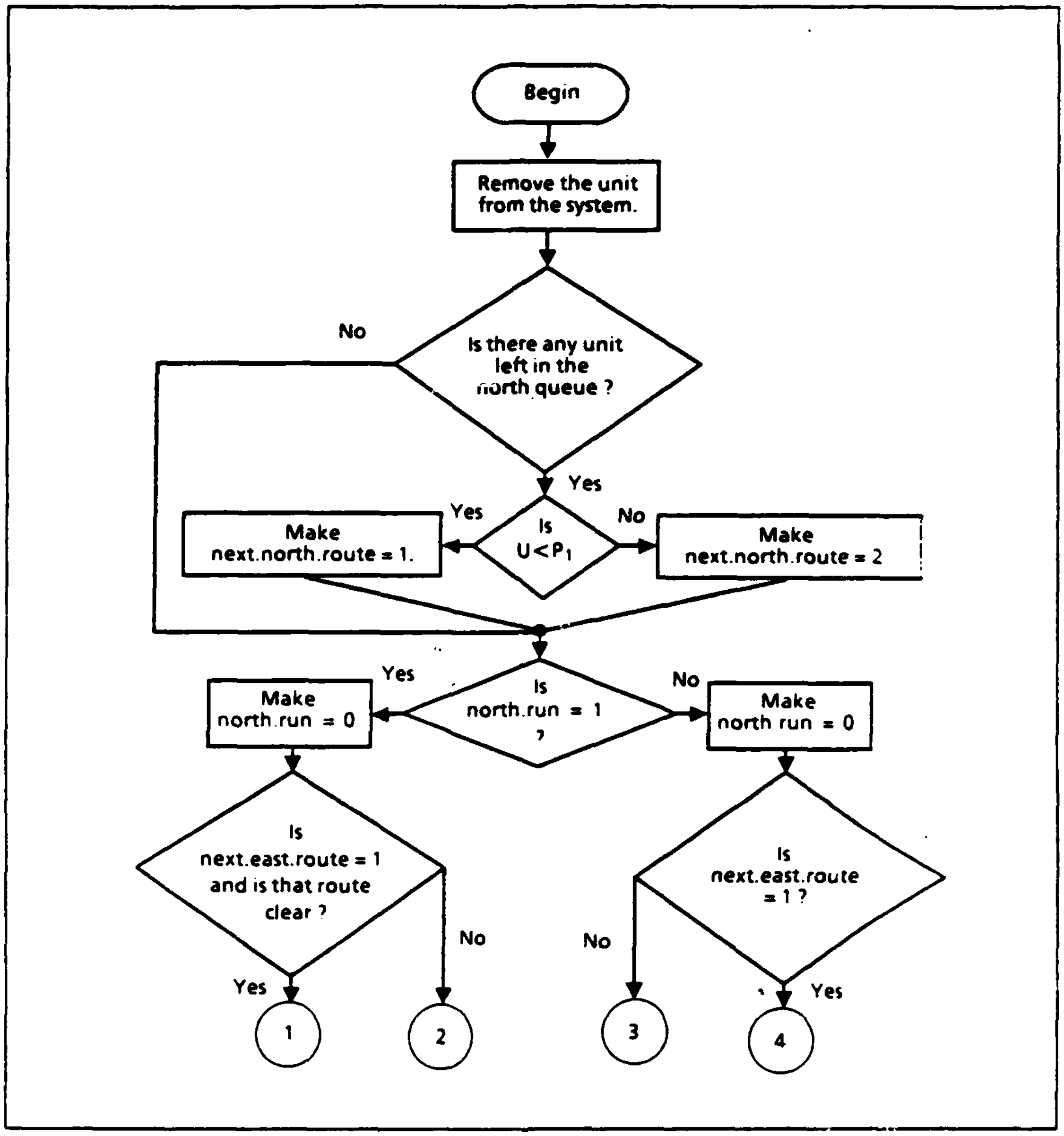

Figure 5. **Flow chart for north.service event routine:** U is a Uniform (0.1) random variate and P_1 is the probability of taking route1. In this routine, priorities for service after a service completion on north route1 or 2 are given to appropriate routes according to the service policies discussed in Table 2 through Table 7 in Chapter II.

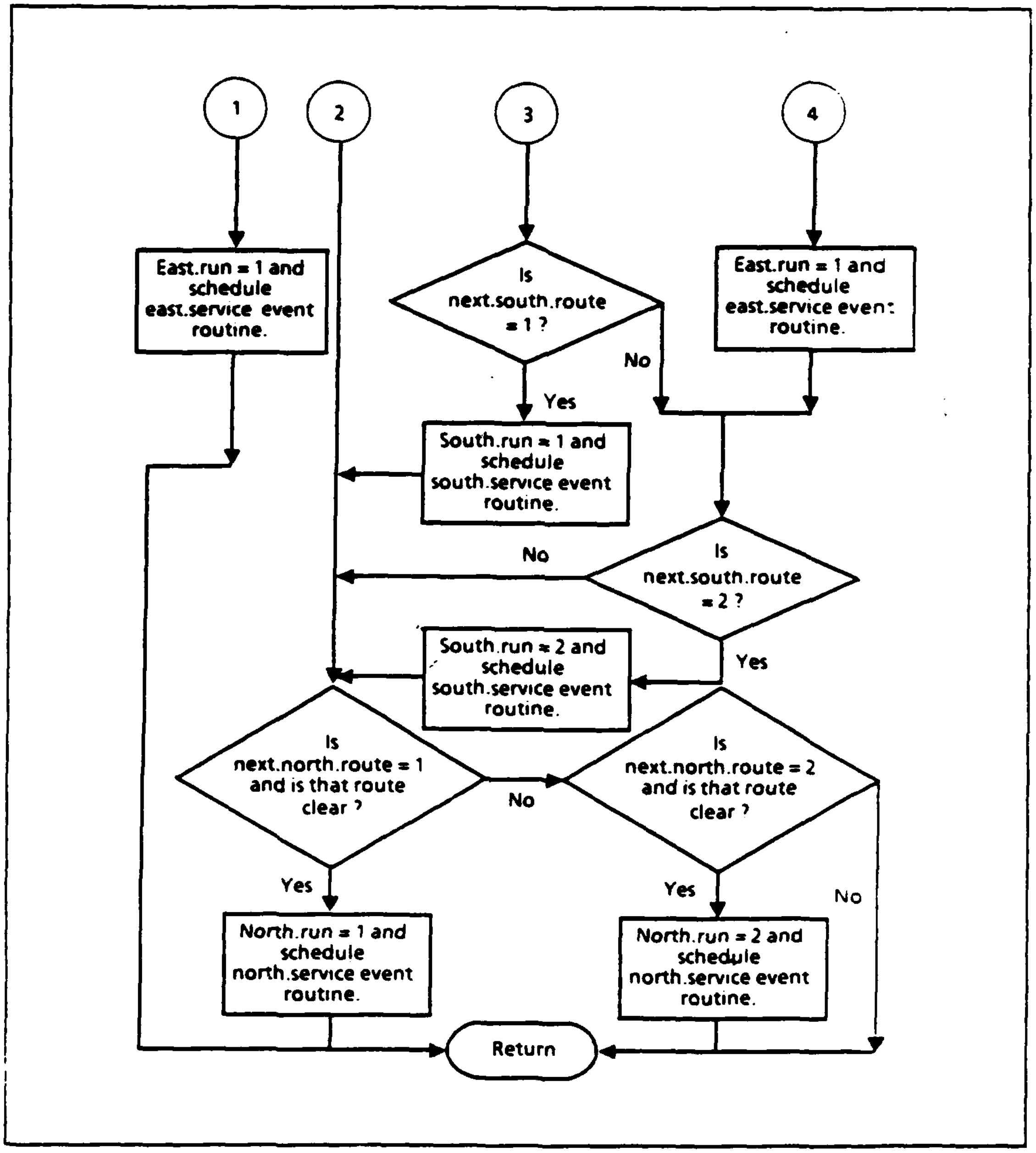

Figure 6. Flow chart for north.service event routine (continued): The priorities for next service after service completion in the south or east are decided in the south(east).service routines. Note that in the program the value of next.north(south,east).route is changed to 0 whenever a north(south,east).service event routine is scheduled.

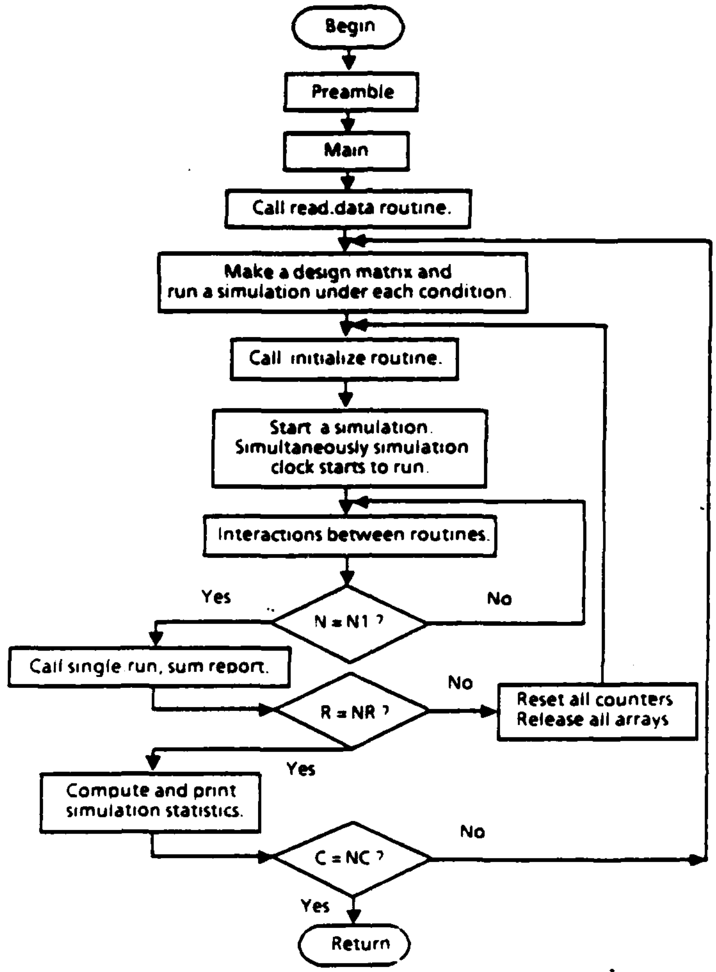

Figure 7. **A macro flow chart for the model:** N is the smallest number of north(south,east) units that have passed the intersection and N1 (selected by the user) is the desired number of units to be simulated. R and NR indicate the number of replications completed and the desired number of replications respectively. C and NC are used to check if all simulation conditions are examined where C indicates the number of conditions that have been examined, and NC is the total number of conditions to be examined. See Table 8 for the simulation conditions.

D. RESULTS

1. Statistical Considerations

When the combinations of many different input specifications to a simulation are to be evaluated, as in this problem, we must first decide upon which combinations of factors we wish to spend our limited simulation resources. So, many preliminary experiments using various combinations of input parameters were investigated, and those conditions under which the system was not stable were used as boundary conditions in the experimental design. As a result, a finite number of discrete level experiments were conducted as a series of essentially independent tests in looking for trends in each direction. We used 100 replications within each run to achieve sufficient precision for each response estimate.

In order to reduce the variance among simulation runs, common random number sequences were used for all the simulations of different factor combinations, [Ref. 11: pp.250-252]. Three separate sequences were used for each simulation (interarrival time, service time, and route selection). The random number sequences were continued to the next replication, so that there was no overlap of the random number sequences and we could therefore assume the replications for each set of factors were independent of each other.

2. Simulation Outputs

Whenever we have one replication of a simulation, as discussed before, a particular realization of the output sequences will be generated. Examples of random sequences of interest would be a running average of successive waiting times and a running average of queue length experienced by the first r north(south,east) units for r = 1,...,N1, where N1 is the desired number of units to be simulated from each direction. N1 is a user input and must be large enough to provide stability.

For this study, the following sequences of interest were collected and computed in the single.run.report and sum.report routines.

- $W_{i,r}$: Waiting time for the r^{th} departing unit from the i^{th} direction, where i can have values north, south or east.
- $T_{i,r}$: Departure time for the r^{th} unit from the i^{th} direction.
- $Q_{i,r}$: The number of units waiting in the i^{th} direction at the r^{th} unit's departure.
- $AW_{i,r}$: The running average of the first r waiting times obtained by the formula:

$$AW_{i,r} = \frac{1}{r}\sum_{v=1}^{r} W_{i,v}, \quad i = \text{north,south,east and } r = 1,...,N1. \tag{3.1}$$

$AQ_{i,r}$: The running average of the number of units waiting in the i^{th} direction during the time $0 \le t \le T_{i,r}$. It is a function of time and a random sequence, but it can also be viewed as a sample of random processes defined on continuous time or as a function or integral of such a process. For example, if we let $Q_i(t)$ be the number of units waiting in the i^{th} direction's queue at time t, then $\{Q_i(t); t \ge 0\}$ is a random process with a continuous parameter t. For $t \ge 0$, $AQ_{i,r}$ can be expressed by the following formula, [Ref. 11: p.272-273]:

$$AQ_{i,r} = \frac{1}{T_{i,r}}\int_0^{T_{i,r}} Q_i(\mu)d\mu, \quad i = \text{north,south,east and } r = 1,...,N1. \tag{3.2}$$

which is easily computed by the ACCUMULATE function in Simscript II.5, [Ref. 7: p.272-273].

The sequences of $W_{i,r}$, $AW_{i,r}$,$AQ_{i,r}$ and the process $Q_i(t)$ for the north direction and 100 units of N1 with skewed interarrival and service time distributions, traffic intensity$=0.5$ and $P_1 = 0.75$ are shown in Figure 8. Note that this set of simulation conditions is also used for further system output displays.

3. Characteristics of Output Sequences

As we discussed above, the outputs of a simulation are sequences of random variables. In this study we are interested in certain probability distribution characteristics and our objective is to estimate them via this simulation since they are unknown and can not be computed with theoretical analysis.

Assume we have M independent observations of the sequence $\{AW_{i,r}\}$ and $\{AQ_{i,r}\}$, $i =$ north,south,east and $r = 1,...,N1$ through independent replications of a simulation. To obtain independent replications of a simulation, we generate independent sequences of random variables to drive the simulation. These sequences are recursively generated, as discussed previously in this section.

To estimate the mean values and obtain confidence intervals of the observations from sets of sample sequences (such as depicted in Figures 9 and 10) we let $\mu_{AW_{i,r}}$ and $\sigma^2_{AW_{i,r}}$ be the mean and variance sequences of $AW_{i,r}$ and $\mu_{AQ_{i,r}}$ and $\sigma^2_{AQ_{i,r}}$ be those of $AQ_{i,r}$, i.e.,

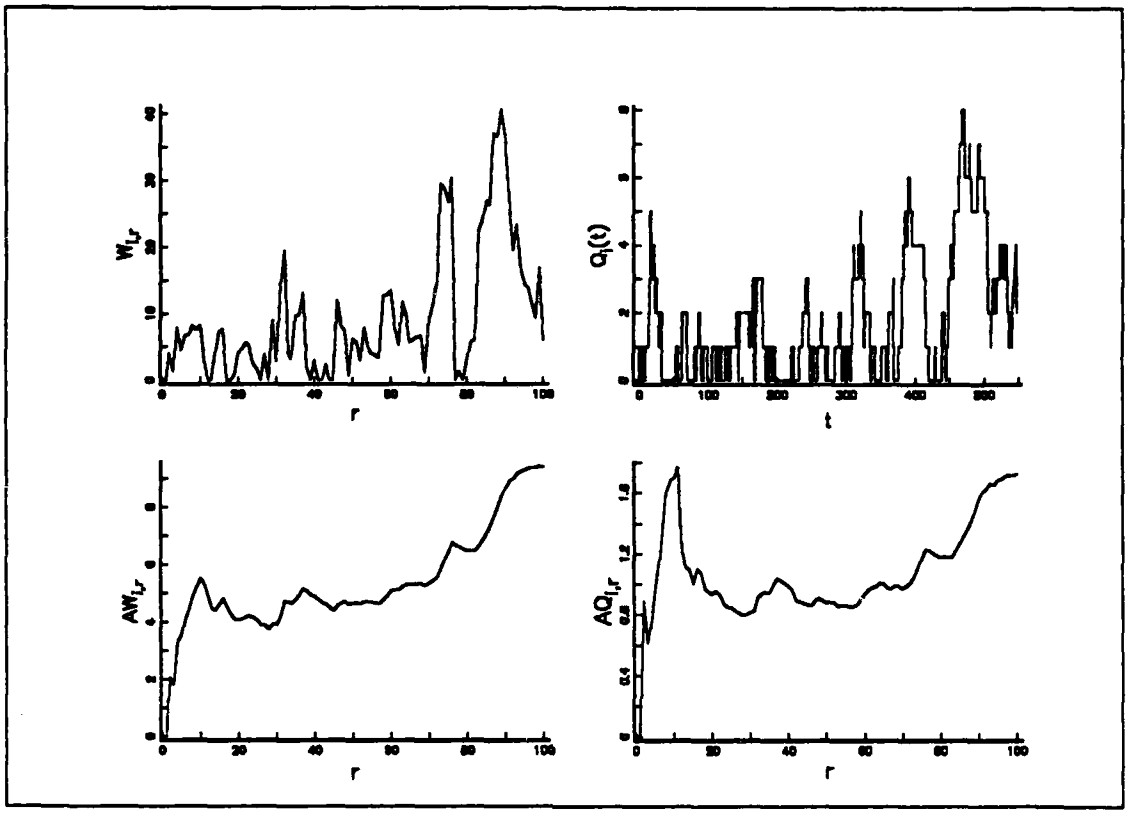

Figure 8. Output sequences obtained from a single simulation run

$$\mu_{AW_{i,r}} = E[AW_{i,r}], \qquad i = \text{north,south,east and } r = 1,\ldots,N1. \tag{3.3}$$

$$\mu_{AQ_{i,r}} = E[AQ_{i,r}], \qquad i = \text{north,south,east and } r = 1,\ldots,N1. \tag{3.4}$$

$$\sigma^2_{AW_{i,r}} = E[(AW_{i,r} - \mu_{AW_{i,r}})^2], \qquad i = \text{north,south,east and } r = 1,\ldots,N1. \tag{3.5}$$

$$\sigma^2_{AQ_{i,r}} = E[(AQ_{i,r} - \mu_{AQ_{i,r}})^2], \qquad i = \text{north,south,east and } r = 1,\ldots,N1. \tag{3.6}$$

$$\hat{\mu}_{AW_{i,r}} = \overline{AW}_{i,r} = \frac{1}{M}\sum_{m=1}^{M} AW_{i,r,m}, \qquad i = \text{north,south,east and } r = 1,\ldots,N1. \tag{3.7}$$

Thus, the sequence of standard point estimators of $\mu_{AW_{i,r}}$ is the sequence of sample means $\hat{\mu}_{AW_{i,r}}$; (3.7) where M is the desired number of replications of the simu-

lations and $AW_{i,r,m}$ is the value of the m^{th} replication of $AW_{i,r}$. A confidence interval for the sequence $\mu_{AW_{i,r}}$ is provided with the t-statistic as follows, [Ref. 11: pp.282-283].

$$S^2_{AW_{i,r}} = \hat{\sigma}^2_{AW_{i,r}} = \frac{1}{M-1}\sum_{m=1}^{M}[AW_{i,r,m} - \overline{AW}_{i,r}]^2, \tag{3.8}$$

$i =$ north,south,east and $r = 1,\ldots N1$ and $M =$ number of replications

$$S_{AW_{i,r}} = \hat{\sigma}_{AW_{i,r}} \quad i = \text{north,south,east and } r = 1,\ldots,N1. \tag{3.9}$$

$$Var[\hat{\mu}_{AW_{i,r}}] = \frac{S^2_{AW_{i,r}}}{M}, \quad i = \text{north,south,east and } r = 1,\ldots,N1. \tag{3.10}$$

Given the above definitions a random variable,

$$\frac{\hat{\mu}_{AW_{i,r}} - \mu_{AW_{i,r}}}{\dfrac{S_{AW_{i,r}}}{M^{1/2}}}$$

has approximately a t-distribution with $M-1$ degrees of freedom and we have

$$Prob\{t_{M-1}(\alpha/2) \leq \frac{\hat{\mu}_{AW_{i,r}} - \mu_{AW_{i,r}}}{\dfrac{S_{AW_{i,r}}}{M^{1/2}}} \leq t_{M-1}(1-\alpha/2)\} = 1-\alpha, \tag{3.11}$$

where $i =$ north,south,east, $r = 1,\ldots N1$, $M =$ number of replications, and $t_{M-1}(\alpha/2)$ is the $\alpha/2$ quantile of the t-distribution with $M-1$ degrees of freedom. Since the t-distribution is symmetric about zero, we obtain,

$$Prob\{\hat{\mu}_{AW_{i,r}} - t_{M-1}(1-\alpha/2)\frac{S_{AW_{i,r}}}{M^{1/2}} \leq \mu_{AW_{i,r}} \leq \hat{\mu}_{AW_{i,r}} + t_{M-1}(1-\alpha/2)\frac{S_{AW_{i,r}}}{M^{1/2}}\} \tag{3.12}$$

$$\approx 1-\alpha, \quad i = \text{north,south,east and } r = 1,\ldots,N1.$$

Therefore the two values, $\hat{\mu}_{AW_{i,r}} - t_{M-1}(1-\alpha/2)\frac{S_{AW_{i,r}}}{M^{1/2}}$ and $\hat{\mu}_{AW_{i,r}} + t_{M-1}(1-\alpha/2)\frac{S_{AW_{i,r}}}{M^{1/2}}$ are 100(1 - α)% confidence intervals for $\mu_{AW_{i,r}}$. Given the previously discussed parameter

conditions, the realizations of $\hat{\mu}_{AW_{i,r}}$ and the confidence intervals of $\mu_{AW_{i,r}}$ obtained from the 6 sequences in Figure 9 and Figure 10 are depicted in Figure 11 and 12 for the north and south directions respectively.

Let $\hat{\mu}_{AW_{i,500}}$ be the 500^{th} value of the sequence $\hat{\mu}_{AW_{i,r}}$; the mean estimator of the running average of successive waiting times experienced by the first 500 units and let $\hat{\sigma}_{AW_{i,500}}$ be the standard deviation estimator of $\hat{\mu}_{AW_{i,500}}$. The values of $\hat{\mu}_{AW_{i,500}}$, $\hat{\sigma}_{AW_{i,500}}$ obtained from various simulation conditions are shown in Tables 16 through 18 of Appendix A. The values of $\hat{\mu}_{AW_{i,500}}$ for different parameter sets and different directions will be analyzed in detail in Chapter V.

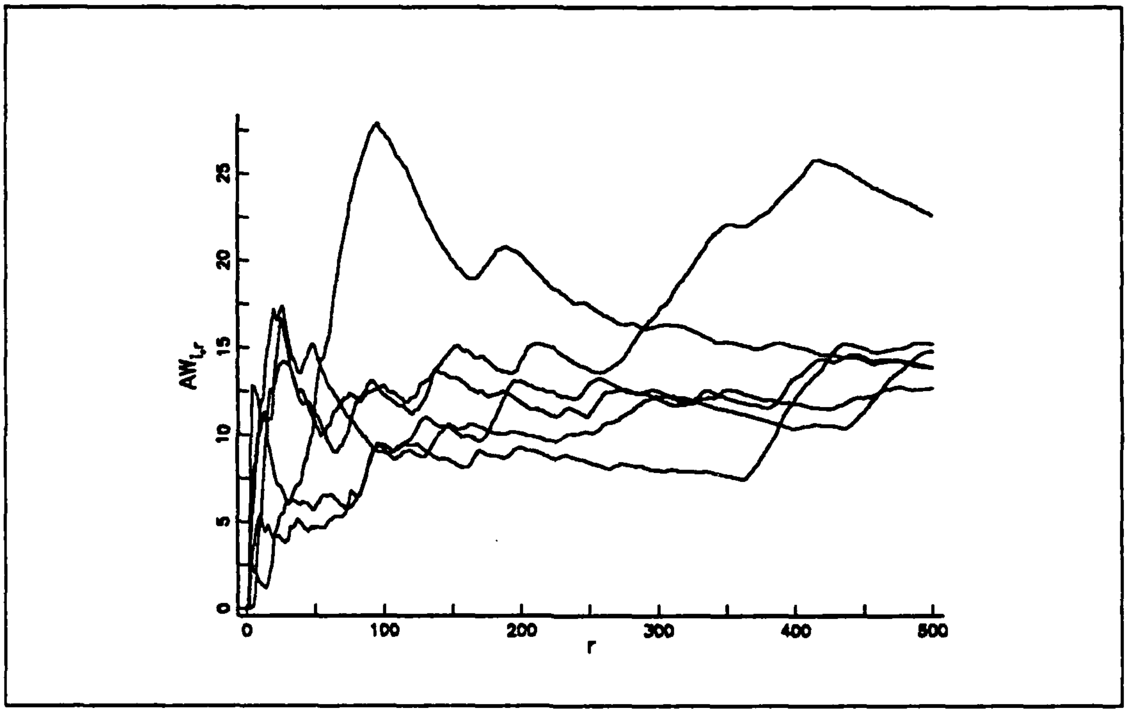

Figure 9. Six realizations of an output sequence (north queue): This figure shows six realizations of the sequence $\{AW_{i,r}\}$ for value i corresponding to the north direction. We can see the random appearance and the variation from sequence to sequence. The distribution characteristics of the random sequences such as these are what we are interested in estimating. These characteristics are estimated from the sequences in the figure and are shown in Figure 11.

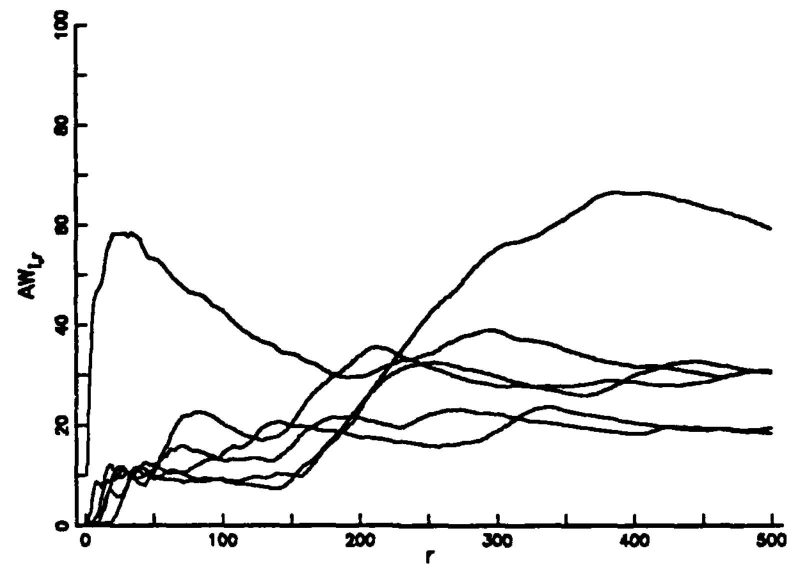

Figure 10. Six realizations of an output sequence (south queue): This figure shows six realizations of the sequence $\{AW_{i,r}\}$ for value i corresponding to the south direction. The sequences in this figure also show random appearance and variation from sequence to sequence. The distribution characteristics of the random sequences are estimated and shown in Figure 12.

4. Stability and Convergence Considerations.

Note that there are three separate queues in the system. As we see in Figure 11 and Figure 12, the sequence $\{\hat{\mu}_{AW_{i,r}}\}$ seems to converge to $\mu_{AW_{i,r}}$ in the north and south queues under the given traffic conditions. We found out that the east queue is not stable under the same conditions. We know that it makes a difference in simulation difficulty if the sequence of interest settles down and converges to a stationary distribution after an initial transient phase. If the sequence settles down, then we may only need to consider the distribution that exists after the transient phase and not a separate distribution for each value in the sequence, [Ref. 6: p.III.22].

To test convergence to the stationary distribution, we made preliminary, independent replications of the simulation over a range of r (number of units simulated)

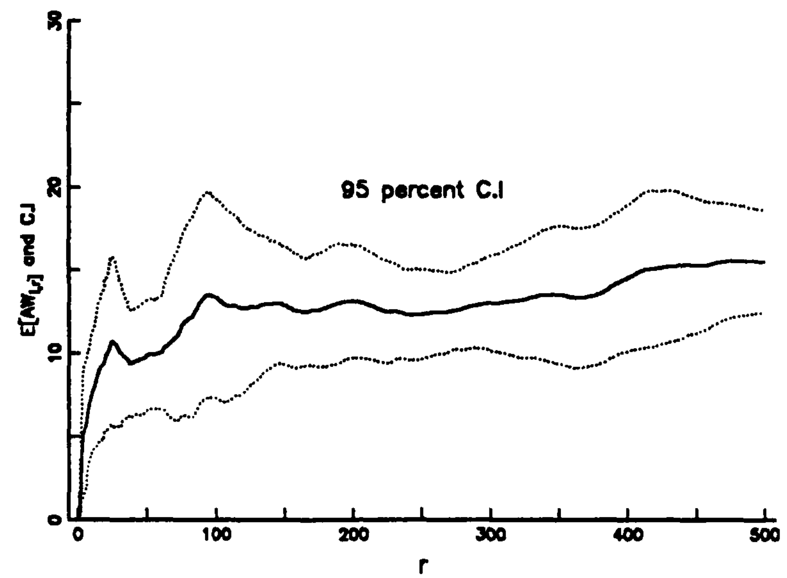

Figure 11. Mean and C.I sequences of waiting time (north queue): This figure shows a realization of $\{\hat{\mu}_{AW_{i,r}}\}$ and confidence intervals of $\{\mu_{AW_{i,r}}\}$ for value i corresponding to the North direction. The sequence $\{\hat{\mu}_{AW_{i,r}}\}$ is a random sequence obtained from 6 sequences of $AW_{i,r}$ in Figure 9. Since $E[\hat{\mu}_{AW_{i,r}}] = \mu_{AW_{i,r}}$, it fluctuates about the converging sequence of means. The larger the value of M the narrower the fluctuations and the closer the approximation to the sequence $\{\mu_{AW_{i,r}}\}$. Hence, by taking a large value M (100 in this thesis) a reasonable estimate can be made on a convergence point for the sequence $\{\mu_{AW_{i,r}}\}$. Notice that from this figure a reasonable judgement can be made as to the extent of the transient phase, whereas this would be impossible from any individual output sequence. In this figure a reasonable estimate for the end of the transient phase appears to be about the first 100 waiting times.

sufficient to observe convergence. The sequence $\{\mu_{AW_{i,r}}\}$ was estimated by generating a sequence of sample means as discussed before. The most straightforward convergence test procedure would be to collect and plot the whole sequence and check the convergence by examining the trends. In practice, however, such an investigation requires more

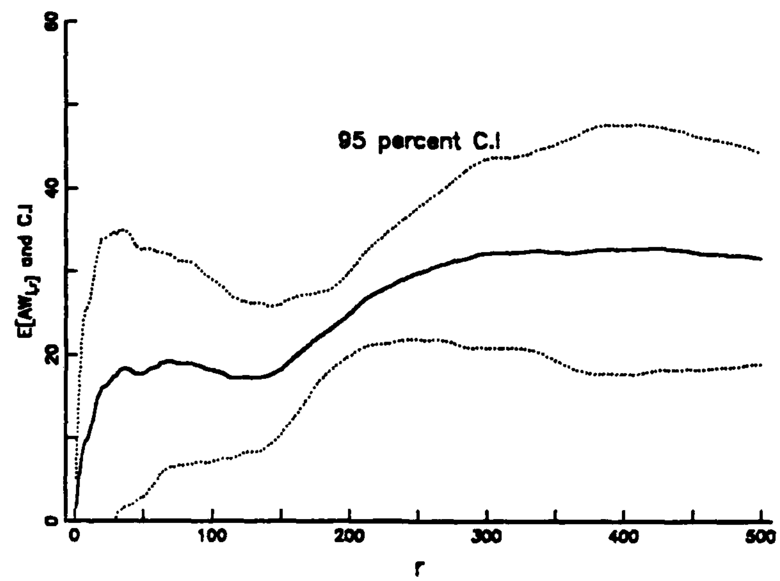

Figure 12. Mean and C.I sequences of waiting time (south queue): This figure shows a realization of $\{\hat{\mu}_{AW_{i,r}}\}$ and confidence intervals of $\{\mu_{AW_{i,r}}\}$ for value i corresponding to the *south direction*. The sequence $\{\hat{\mu}_{AW_{i,r}}\}$ is a random sequence obtained from 6 sequences of $AW_{i,r}$ in Figure 10. In this figure we can see that the south queue is stable and a reasonable estimate for the end of the transient phase appears to be about the first 300 waiting times.

computing time and effort than would be reasonable. Therefore we collected every 10^{th} data point after the first 300 points of a sample mean sequence, such as the one seen in Figure 11, to save our limited simulation resources. To examine convergence of the resulting sequence we manually compared the values over the sequence range.

The convergence tests revealed the system is partially stable in the north and south queues for Model 1 under the condition that traffic intensity = 0.5, $P_1 = 0.75$ and the ratio of mean interarrival time for east over mean interarrival time for north or south is 1. To illustrate the east direction's unstable queue, we collected 6 complete sequences of $AW_{east,r}$ and plotted them in Figure 13. Note the upward trend in the sequences. As we increase the ratio; (mean interarrival time for east) / (mean interarrival time for

north), from 1 to 1.3 by increments of 0.1, the east queue becomes more stable (Figure 14).

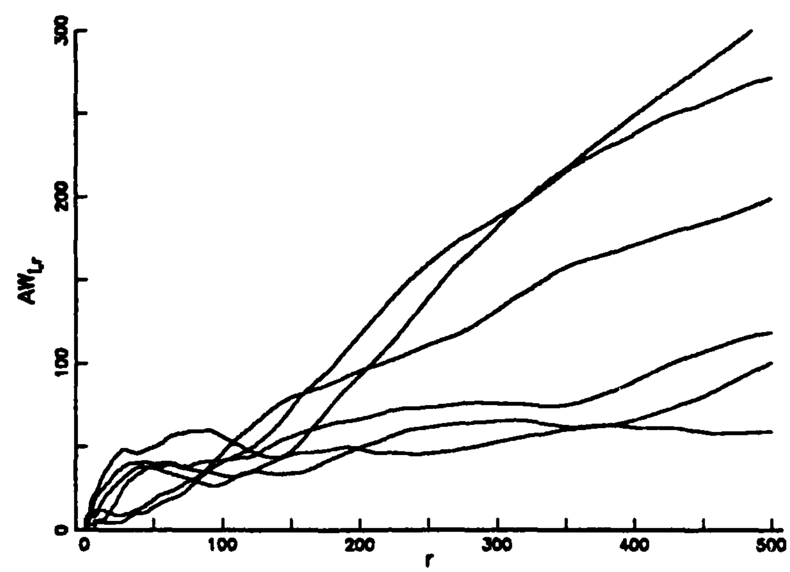

Figure 13. Six realizations of an output sequence (east queue): This figure shows six realizations of the sequence $\{AW_{i,r}\}$ for value i corresponding to the east direction. The sequences show an upward trend and indicate there is no steady state distribution for the mean of the sequences. The characteristics of each data point in the mean sequence can be estimated from the corresponding data points in the six $AW_{i,r}$ sequences.

5. Data Collection and Analysis Tools

We have powerful tools for data collection and analysis such as "print" and "accumulate/tally" statements in PC-simscript. These statements can be used for data collection and some statistical calculations. Other software packages such as STATGRAF in PC can be used for more sophisticated data analysis. In addition to these tools for use on the personal computer, we used IBM 3033AP mainframe support to efficiently manipulate much of the data. Data was collected on floppy diskettes and transferred to the IBM mainframe with the 3270 PC's Host-Supported File Transfer program. Once transferred to the mainframe, we edited the data file prior to analyzing

the data with GRAFSTAT. GRAFSTAT graphical displays were extremely useful for checking the characteristics of large data sets such as required for Figure 9 and Figures 16 through 22 in Chapter V.

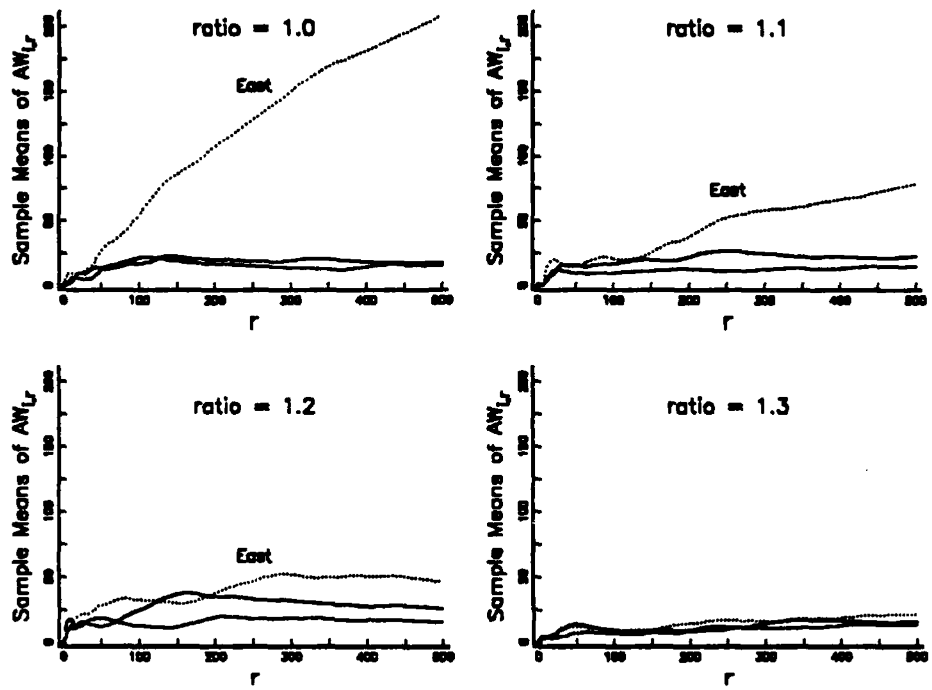

Figure 14. Stability test on the east queue: We tested the stability of the east queue by increasing the ratio of mean interarrival time for the east over the mean interarrival time for the north from 1.0 to 1.3. The estimators of $E[AW_{i,r}] = (\mu_{AW_{i,r}})$ is $\hat{\mu}_{AW_{i,r}}$, the sequence of the sample means of $\{AW_{i,r}\}$ for value i corresponding to the east direction. The east queue becomes more stable as the "ratio" increases.

IV. SIMULATION MODEL 2

A. ASSUMPTIONS AND CHANGES TO THE MODEL

After developing a model of the system using the described assumptions, we propose and simulate a road structure to reduce the restrictions of the current road system. Figure 15 depicts the proposed system which will be addressed as Model 2. It is obvious that Model 2 allows more combinations of routes, so that traffic performance at the intersection should be increased. Actually 4 combinations of routes, (0,1,1), (2,0,1), (2,1,0), and (2,1,1), are added to the 14 cases of Model 1. In case of blockages, the priorities for next service are shown in Table 9. Note that in Model 2, each route only blocks the route that merges with it in a common direction. All other assumptions of Model 2 remain the same as in Model 1.

Table 9. PRIORITIES AT THE END OF SERVICE OF A BLOCKING ROUTE (MODEL 2): See Table 1 and explanation of Table arrangement provided on page 7 for comparison with Model 1.

Route which is about to finish service	Next Priority, in order, at the end of blocking
north route1	east route1, (north route1, north route2)
north route2	south route2, (north route1, north route2)
south route1	east route2, (south route1, south route2)
south route2	north route2, (south route1, south route2)
east route1	north route1, (east route1, east route2)
east route2	south route1, (east route1, east route2)

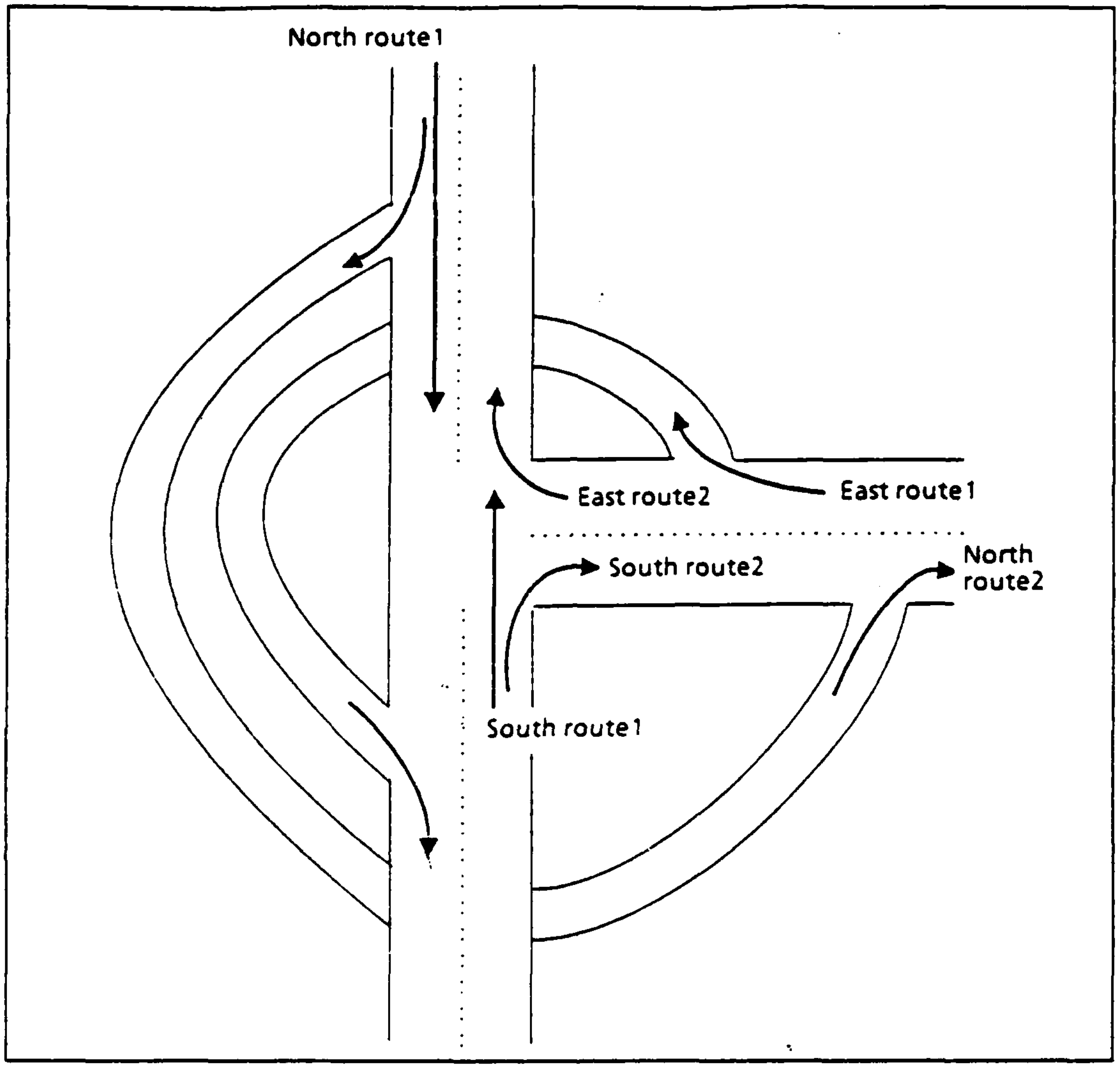

Figure 15. **A proposed modification of the current system (Model 2):** Here several blockages have been removed. In particular traffic coming from east route1 is never blocked by traffic coming from south route1 though it still has to merge with traffic from north route1. Similarly, traffic on north route2 is no longer blocked by either traffic on east route1 or south route1, though it still can be blocked by traffic from South route2. Finally, traffic on south route1 is no longer blocked by traffic on north route2 or east route1 but can still be blocked by traffic from east route2.

B. INPUTS TO THE MODEL

Inputs are the same as in Model 1.

C. MODEL IMPLEMENTATION

Since the only difference between Model 1 and Model 2 are the additional route combinations, all other events and routines remain the same as the first model except the events related to evoking service. These routines are new.north(south,east).unit and north(south,east).service event routines. As an example, consider the route combination (2,1,0); it is not allowed in Model 1 because of the movements of north route2 and south route 1 are in conflict, however, the two movements are allowed in Model 2. Thus the decision phase of the event routines are revised for the new model as required to allow the additional route combinations. The program listing is shown in Appendix C.

D. MODEL VERIFICATION AND VALIDATION

Having set up the service policies, system assumptions and developed the simulation model, the next step is to verify the model; that is, to detect and correct discrepancies between the intended and actual execution processing performed by the model.

A simulation model of a complicated process such as this model can not be debugged or verified unless comprehensive diagnostic outputs, which completely describe the operation of the simulated system, are provided. Diagnostics are needed only for debugging, being much too cumbersome and costly to use in full-scale runs. The most powerful and economic diagnostic tool used in this model is output which describes each simulated event on a separate line with almost all relevant information about each unit's status. The data collected in the single.run.report routine is also used for verification of the model. The sample sections of this diagnostic and the data collected in the single.run.report routine are shown in Tables 13 through 15 of Appendix A. Another verification tool used was to examine the logic of the flow charts discussed earlier.

To validate the model as a sufficiently accurate representation of the intersection operations for the purpose of the study, that is, to answer the question of whether there is a correspondence between the model and the physical intersection system, would require the models results be compared with that of the real system in a given situation. It would be impossible to create the physical system conditions simulated by the model because of the many restricted military situations in R.O.K. We can not exercise the movements of all forces to simulate wartime traffic given the current situation. In spite of the restrictions, manual checks of the model flow charts and careful examination of the assumptions and service policies may be conducted to confirm that the model cor-

responds to the actual system. If we assume that military police control the traffic at the three way intersection, the general assumptions and service policies in the models do not correspond to the real world, however, that is not a problem since we are interested in overall traffic performance at the intersection and service priority issues do not affect our study objective.

V. ANALYSIS

A. SIMULATION RESULTS

1. Current Road System (Model 1)

We present a graphical approach in organizing and presenting the simulation results, [Ref. 6: p. VIII.3-23]. The interest is showing, in a simple graphical form, how the chosen quantification, i.e. mean waiting times at the intersection, vary with factors such as traffic intensities, service(unit) length and interarrival time distributions. Figures 16 through 18 show the results of the first model with factor $P_1 = 0.75$, 0.50 and 0.25 respectively.

In Figure 16, we have six sub-figures; three in the upper half with mean service times of 2.5 for all directions and three in the lower half with mean service times of 5.0 for all directions. Each cell in the sub-figure is indexed by the traffic intensity and the combinations of interarrival and service time distributions defined here to be SS, SR, RS, and RR, (see Figure 16 for an explanation of these terms) and contains the estimated value of $\mu_{AW_{i,500}}$ (see Equations 3.7 and 3.9). In Figures 17 and 18, we can observe mean waiting times of the three directions for all combinations of interarrival and service time distributions, given traffic intensity = 0.5 and mean service time = 2.5. In addition, Figure 22 shows the changes to the mean waiting times of the east queue together with changes to the mean interarrival time ratios for the three directions, given Model 1 with $P_1 = 0.75$, traffic intensity = 0.5, and mean service time = 2.5. Note that the waiting times in Figures 17 and 18 should be compared with the waiting times on the graphs in Figure 16 with mean service time 2.5 and traffic intensity of 0.5. Note too that the vertical scales in Figures 16 through 18 are different. It would be preferable to use the same scale to give a depiction of the relative difference of the mean waiting time as simulation parameters change, however, this would require either a very large display or a transformation of the data (such as a log scaling). Neither of these were desirable, thus we need to look at the vertical scales when comparing the estimators between sub-figures.

Hence, it is relatively easy to visually inspect Figure 16 to determine the effect of the mean service time (by looking at the figure's columns), the effects of traffic intensity (by comparing the rows in a sub-figure), and the effect of the combinations of interarrival and service time distributions (by looking at the columns in a sub-figure).

However, in order to observe the effects of changes to the probability that the first unit in the queue selects route1, P_1, we need Figures 16 through 18 side-by-side.

It is immediately clear in Figures 16 through 18 that traffic intensity ρ has a major effect on the mean waiting time and, in fact, it is known from theory that the mean waiting time in many situations is proportional to the reciprocal of $(1-\rho)$, [Ref. 6: p.VIII.8]. Beyond this, it appears to be generally true that the regular-regular combination of interarrival and service times at the right most position in the six skyscraper plots of Figure 16 is best for traffic performance; on the other hand, the skewed-skewed combination at the left most position in the six plots is the worst case for all values of traffic intensity. In addition, as the mean service time doubles the mean waiting time also doubles. Finally, the effects of increasing mean service time from 2.5 to 5.0 together with other factor combinations turns out to be consistent with different values of P_1.

To analyze the effects of the factor P_1 requires comparing Figure 16 ($P_1 = 0.75$) through Figure 18 ($P_1 = 0.25$). For instance, the mean waiting time of the east direction is almost five times that of the other directions in Figure 16 with $P_1 = 0.75$. On the other hand, Figure 17 with $P_1 = 0.5$ shows that the mean waiting times for the south direction are relatively higher than the north and east directions. This is because the frequency of movements on each route varies as the value of P_1 changes in conjunction with different route movement conditions.

Using Figure 22 for $P_1 = 1$ and traffic intensity = 0.5, we found that the system is stable in the north and south directions but can be unstable in the east direction if

$$\frac{\text{mean interarrival time of east units}}{\text{mean interarrival time of north or south units}} \leq 1.3.$$

A more sophisticated analysis of the results, for instance the test to see if there still remains a difference in mean waiting times of three directions due to P_1 after adjusting other factors, is not done in this paper, [Ref. 6: p.VIII.9].

2. Proposed Road System (Model 2)

The results of the simulation for Model 2 are shown in Figures 19 through 21 and the waiting times in these figures are compared with the corresponding values of Figures 16 through 18. The effects of the factors found in the first model are consistent in this model. We can expect that the overall traffic performance in Model 2 would increase because the new system allows more combinations of movements, that is, the restrictions to movements are reduced. The benefits of the new system for the north direction removes two possible blocking routes, east route1 and south route1, which are

blocking units in the first model for north route2. As for the south direction, east route1 and north route2 no longer block south route1. Finally in the case of the east direction, east route1 is no longer blocked by south route1 or north route2. Thus, overall movement conditions are improved and waiting times, as would be expected, are decreased for the three directions in this model.

B. COMPARISON OF THE TWO MODELS

In order to compare the results of the two simulations, the sample means and standard deviations of the first 500 average (running) waiting times for each direction (see equations 3.7 and 3.9) are tabulated in Tables 19 through 21 of Appendix A, so that a direct comparison of the waiting times for the two models is possible on a pair-wise basis. The fact that the new model is more efficient than the current model in traffic performance is evident. However, the criteria to prefer the new system to the old could be more than just the relative differences in time. Numbers of additional units deployed per unit of time, the decrease in forces susceptible to enemy air and ground ambushes, are a few of the criteria that could be used to compare the two systems.

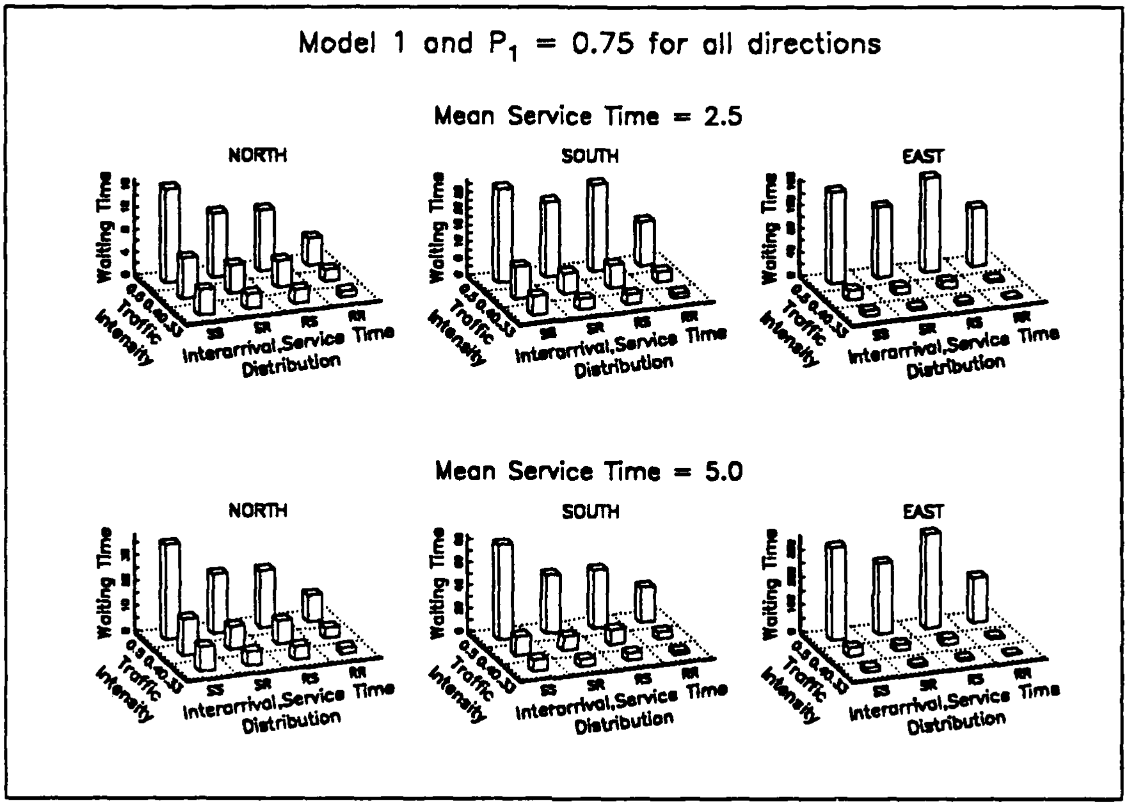

Figure 16. ***Effects of major factors for Model 1:*** Each bar in the cell shows the mean estimate of the first 500 (N1 = 500) waiting times for each direction's queue. The notation SS, SR, RS, and RR on the X axis represent combinations of interarrival and service time distributions where S and R means 'skewed' and 'regular' interarrival or service times respectively. For instance, the notation SR indicates 'skewed' interarrival and 'regular' service time distributions were used for the simulation. These are achieved by using different Gamma shape parameters k_1, k_2 for the interarrival and service times. The effects of the traffic intensity and the combination of interarrival and service time distributions are obvious and consistent in the six skyscraper plots. The mean waiting times in the east direction are very large and unstable while the other two directions are stable. This is discussed further in Figure 22. Note that the scales are different between the six sub-figures.

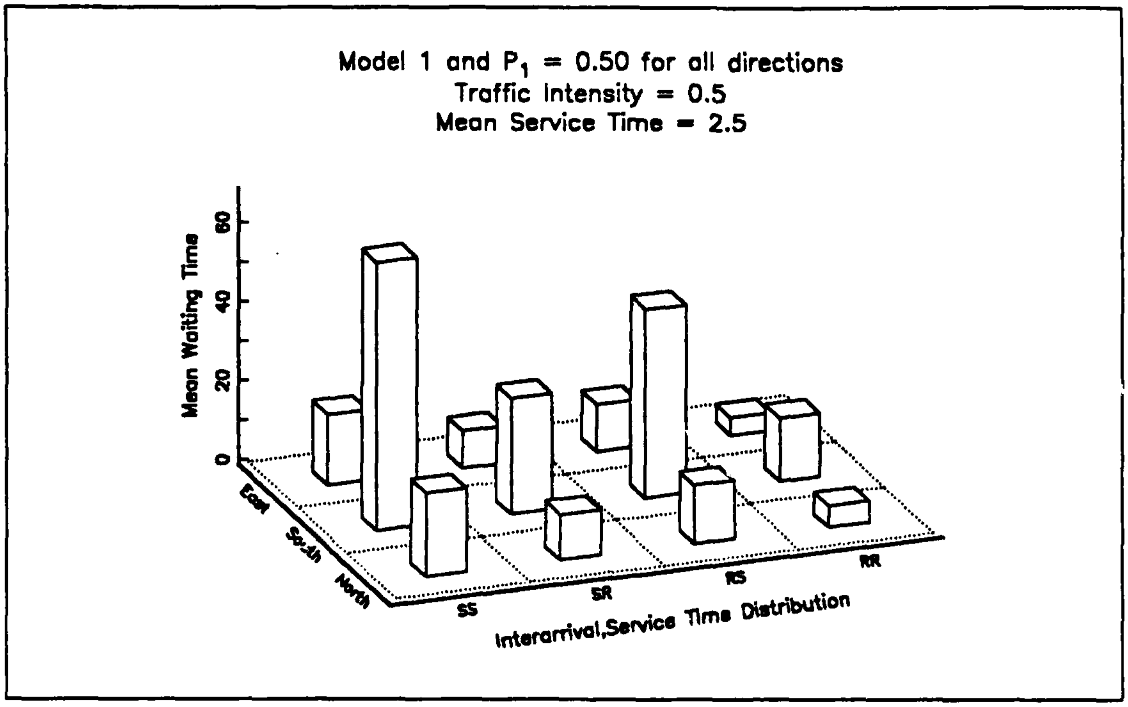

Figure 17. Effects of probability of taking route1(0.50) for Model 1: See Figure 16 for the explanation of SS, SR, RS, and RR in the plot. The mean waiting times in this plot correspond to a mean service time 2.5 and traffic intensity of 0.5 for each direction. The effects of decreasing P_1 from 0.75 in Figure 16 to 0.50 in this figure is to decrease the east's waiting time and to increase the south's waiting time. The decrease in the east's waiting time is because the probability that east units take east route2 increases to 0.50 from 0.25 in Figure 16 and it is easier for east units to move to east route2 than east route1 since east route1 has three blocking routes. On the other hand, there is only one blocking route for east route2. The reason for the increase in south's waiting time is because more north units select north route2 and more east units select east route2 which combine to increase the difficulty for south units to move under $P_1 = 0.50$ vs $P_1 = 0.75$.

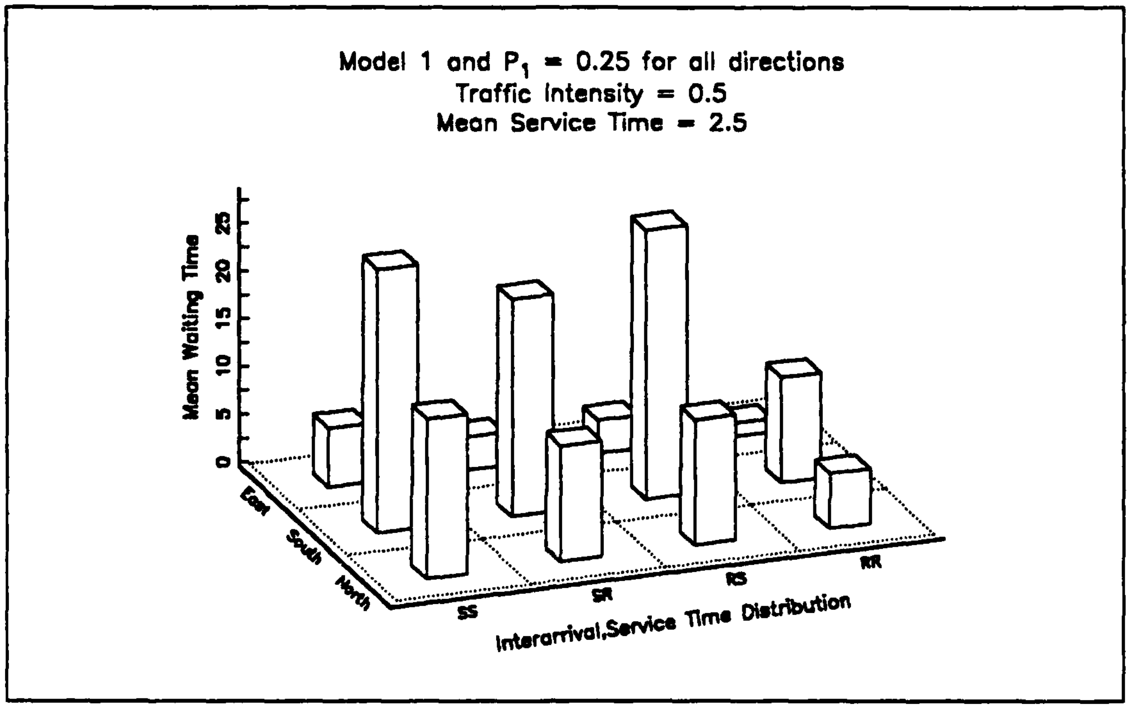

Figure 18. Effects of probability of taking route1(0.25) for Model 1: See Figure 16 for the explanation of SS, SR, RS, and RR in the plot. The mean waiting times in this figure are compared with the corresponding values in the third row of the upper three plots of Figure 16 and against Figure 17. The major effects of decreasing the value of P_1 from 0.50 in Figure 17 to 0.25 in this figure is to decrease east's and south's waiting times. The decrease in east's mean waiting time is because now east units select east route 2 with probability 0.75 and east route2 is only blocked by south route1, which is selected by south units with probability 0.25. That is, the probability that an arriving east unit is blocked becomes lower than in Figures 16 or 17. The decrease in the south waiting times can be contributed to the difference in the probability of taking each route and the interactions of movements on the routes.

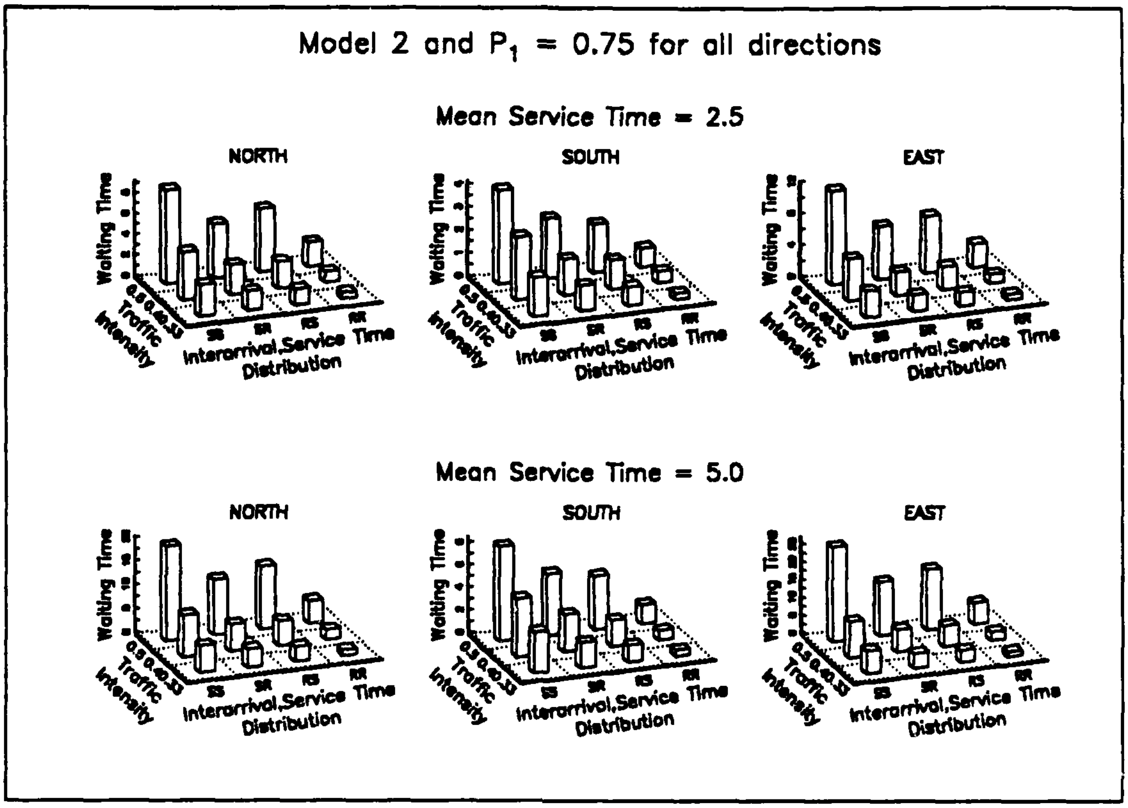

Figure 19. ***Effects of major factors for Model 2:*** See Figure 16 for the explanation of SS, SR, RS, and RR in the plots. The waiting times on the north, south and east directions decreased almost 2, 6 and more than 10 times respectively from the corresponding waiting times of Figure 16 in the first model. The differences in the rates are because the performance benefits of the new system are different for each direction. For example with $P_1 = 0.75$, the benefits of new east route1 and north route2, are different since approximately 75% of the east units use east route1 and only about 25% of the north units use north route2. However, the overall effects of traffic intensity and mean service time remain the same as in Model 1.

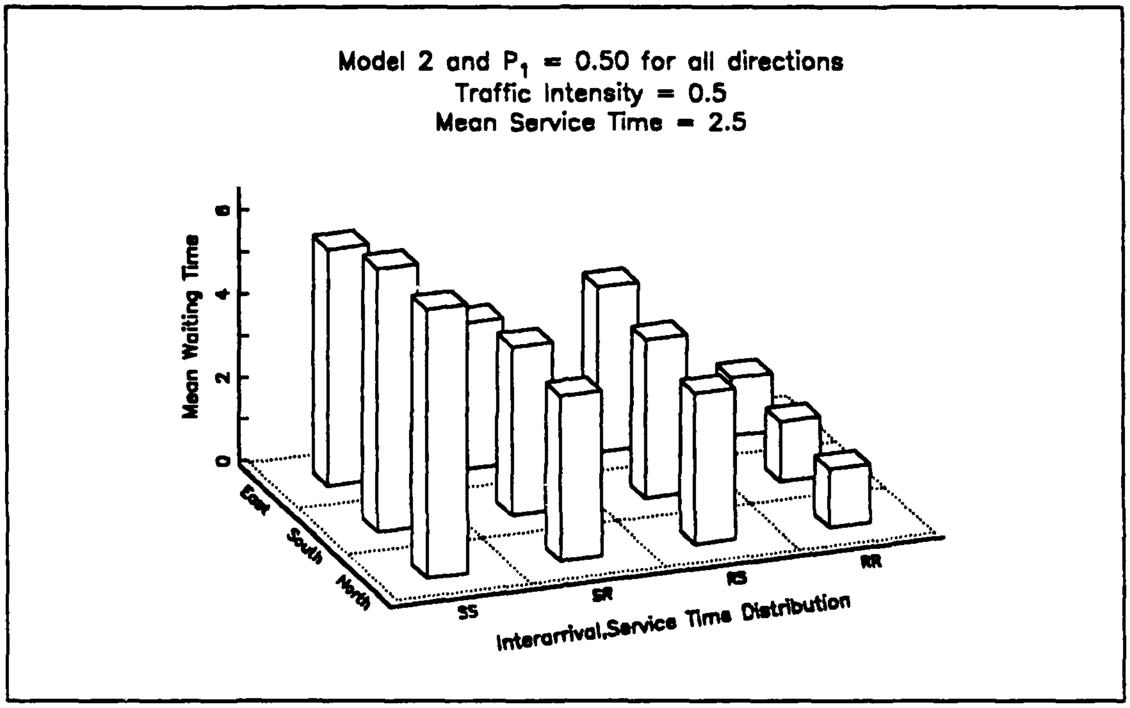

Figure 20. Effects of probability of taking route1(0.50) for Model 2: See Figure 16 for the explanation of SS, SR, RS, and RR in the plot. The mean waiting times in this figure can be compared with the corresponding values in Figure 17 to see Model 2's effect. The corresponding values in the third row of each direction for the upper three plots in Figure 19 depict the effects of changes to P_1 in Model 2. Since each direction has only one blocking route and $P_1 = 0.5$, it is reasonable to expect that the mean waiting times of the three directions are approximately equal.

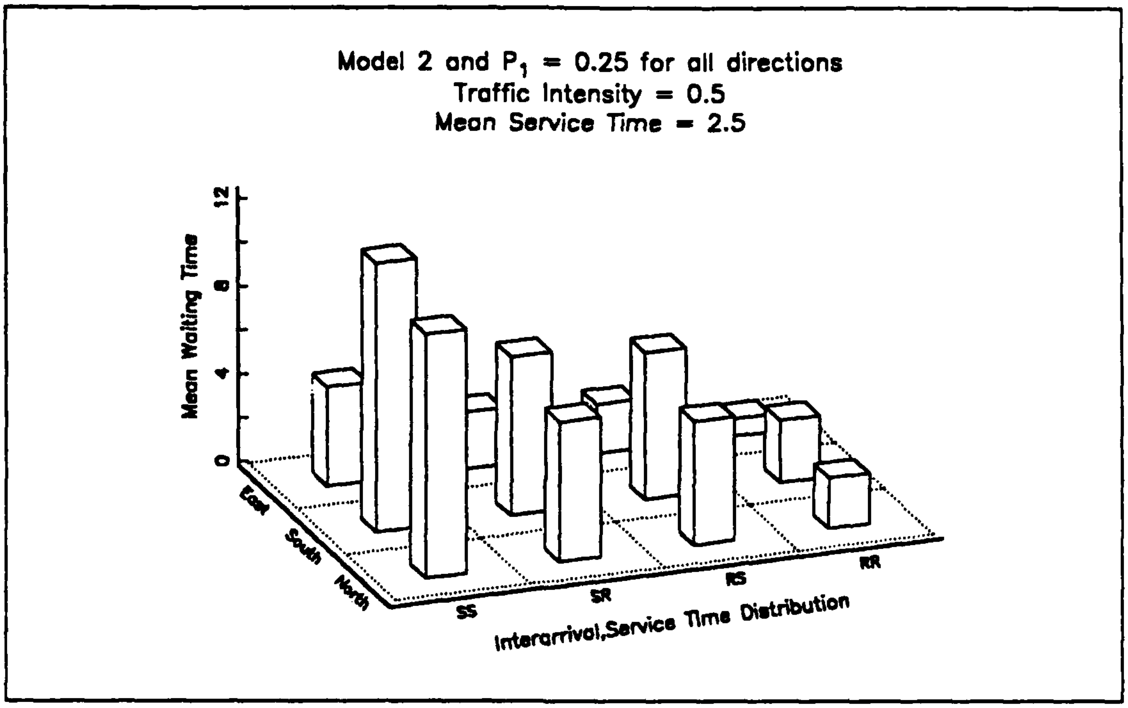

Figure 21. Effects of probability of taking route1(0.25) for Model 2: See Figure 16 for the explanation of SS, SR, RS, and RR in the plot. It is again obvious that the mean waiting times decrease in comparison with the corresponding case of the first model (Figure 18). The south and north have longer waiting times than the east because north and south units select north route2 and south route2 with probability 0.75. On the other hand, east units interact only with the north and south units which take routes blocking east units with probability 0.25.

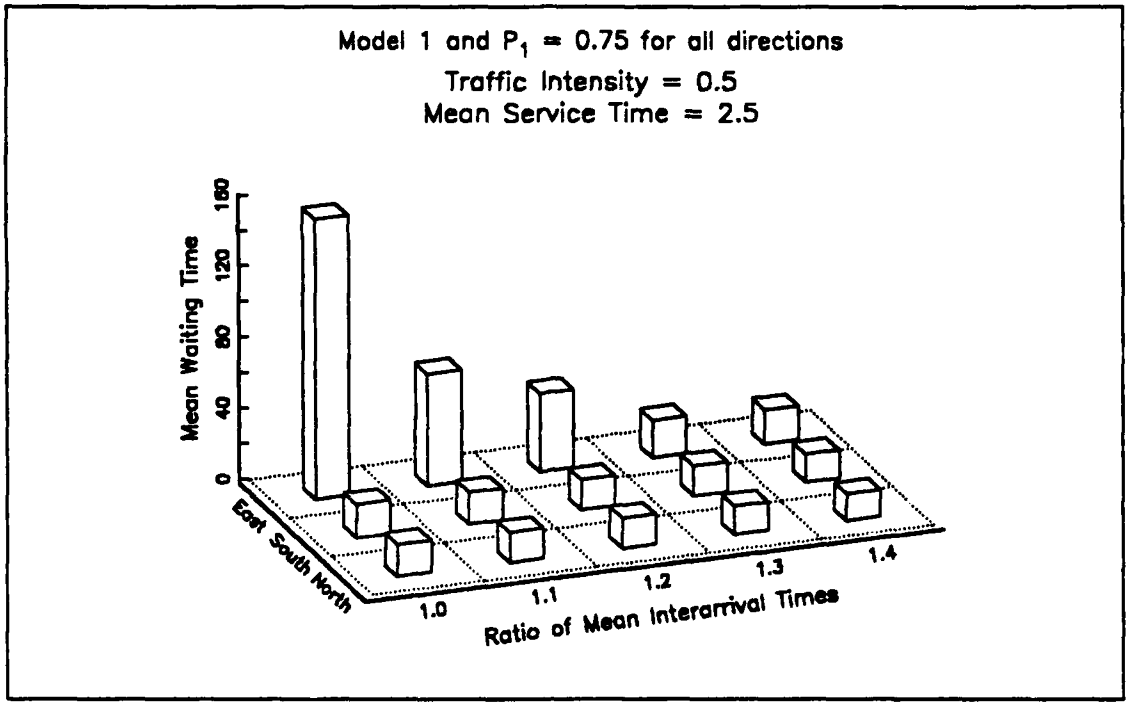

Figure 22. **Stability changes on east queue (Model 1):** This figure shows the specific characteristics of the intersection given Model 1 with $P_1 = 0.75$ and traffic intensity = 0.5. For these given input parameters, this figure shows changes of stability on the east queue given changes to the mean interarrival time ratios for the three directions. The changes are consistent with any combination of interarrival and service time distributions, i.e. SS, SR, RS, and RR (see Figure 16 for an explanation of SS, SR, RS, and RR). The results in this figure are from skewed interarrival and service time distributions (see page 21). The numbers on the X axis; 1.0, 1.1... represent the ratio of (mean interarrival time of east units / mean interarrival time of north or south units). As seen in Figure 16, the east queue is unstable if the arrival rate from the east is the same as that of north and south, and becomes stable as the arrival rate from the east decreases with respect to the north or south arrival rate. North's and south's stability was not affected by the ratio changes.

VI. CONCLUSIONS AND RECOMMENDATIONS

This research has demonstrated the effects of traffic conditions on traffic flow at two kinds of three way intersections. We have seen how traffic along the roads of interest varies as the conditions change. The question of how we interpret mathematical terms such as Gamma distributions and other random variables in the real world still remains uncertain. That is, how do we know actual interarrival and service times would be distributed with certain characteristics? For example, we have assumed a Gamma distribution for interarrival and service times in this model, however this assumption may be completely different from the real world situation.

As for service time, it seems as if many factors affect service in wartime in addition to the physical restrictions of a road system. As mentioned before, major problems such as refugee flow along the roads should be studied separately. But we can assume a certain distribution for service time for the purpose of this study.

For interarrival times, however, we can collect the appropriate data by conducting realistic war game simulation. As the war game situation changes, the target level forces are assumed to complete their movement according to the combat model's movement schedule, [Ref. 12: p. 3.15-23]. The movement speed is assumed to vary according to the characteristics of the theater. However, the current models do not consider the road movement interaction of forces which may create critical delays in combat force operations. Therefore we must consider the road movement interaction of combat forces given various wartime conditions and find out whether the movements of the target level forces in the wargame simulation can be completed in a timely manner in the real world. This study enables us to analyze the delays in combat force operations by considering the interaction of convoys which have to traverse an intersection of interest. If we create a road march plan for the forces according to a war game simulation situation then we can collect the expected arrival times of each unit at the intersection of interest. The expected arrival times at an intersection of interest obtained from this procedure could be developed into a distribution for interarrival times.

Thus we can apply this model to the real world with the given assumptions and can determine expected traffic flows in the area of interest. In this thesis, we study only one three way intersection in an attempt to describe its overall traffic performance. It would be especially useful to apply this model to a three way intersection which is believed to be a traffic choke point in a specific area of interest. Further studies about more ag-

gregated road systems along with war game simulation are needed to further understand the potential for wartime traffic problems in the Korean theater.

APPENDIX A. SAMPLE OUTPUT OF THE SIMULATION MODELS.

Table 10. SAMPLE SECTION OF DIAGNOSTIC PRINTOUT OF MODEL 1: Simulation conditions; $P_1 = 0.75$, mean service time = 2.5, TI(traffic intensity) = 0.5, interarrival and service time distribution = SS (see Table 8 and Figure 16).

NNU: new. north. unit, NSU: new. south. unit, NEU: new. east. unit
NS: north. service, SS: south. service, ES: east. service
U: a uniform random variable used for the next route selection.
Movements: movements at the intersection after the left event.
N,S,E: north, south, and east direction.
Queue: number of units in each queue after the left event.
Next route: routes selected for the first unit in each queue.

Event	Time	U	Movements (N, S, E)	Queue (N,S,E)	Next route (N,S,E)
NNU	.39	.45	1 0 0	0 0 0	0 0 0
NNU	.51	.43	1 0 0	1 0 0	0 0 0
NS	4.62	.66	1 0 0	0 0 0	0 0 0
NEU	7.08	.52	1 0 0	0 0 1	0 0 1
NNU	7.51	.04	1 0 0	1 0 1	0 0 1
NEU	7.88	.44	1 0 0	1 0 2	0 0 1
NSU	8.62	.34	1 1 0	1 0 2	0 0 1
NEU	8.79	.54	1 1 0	1 0 3	0 0 1
NS	8.83	.42	1 1 0	0 0 3	0 0 1
SS	8.96	.23	1 0 0	0 0 3	0 0 1
NNU	9.61	.54	1 0 0	1 0 3	0 0 1
NSU	13.17	.07	1 1 0	1 0 3	0 0 1
NNU	13.62	.82	1 1 0	2 0 3	0 0 1
NNU	14.58	.33	1 1 0	3 0 3	0 0 1
NS	15.33	.26	1 1 0	2 0 3	0 0 1
NNU	15.62	.08	1 1 0	3 0 3	0 0 1
NEU	16.04	.73	1 1 0	3 0 4	0 0 1
SS	16.25	.50	1 0 0	3 0 4	0 0 1
NNU	16.32	.20	1 0 0	4 0 4	0 0 1
NEU	16.66	.57	1 0 0	4 0 5	0 0 1
NS	17.40	.44	0 0 1	4 0 4	1 0 0
ES	17.92	.91	1 0 2	3 0 3	0 0 0
NS	18.13	.31	1 0 2	2 0 3	0 0 0
ES	19.23	.55	1 0 0	2 0 3	0 0 1
NS	19.96	.91	0 0 1	2 0 2	2 0 0

Table 11. SAMPLE SECTION OF SINGLE.RUN.REPORT OF MODEL 1.

AT: arrival time of a unit.
Q1: queue length after an arrival.
Q2: queue length before departure of a unit.
WT: waiting time of each unit.
ST: departure time of a unit.
ET: time of service completion.
DIR: route number used by a unit.

	NO	AT	Q1	WT	ST	ET	Q2	DIR
North	1	.39	0	0.	.39	4.62	0	1
	2	.51	1	4.10	4.62	8.83	1	1
	3	7.51	1	1.32	8.83	15.33	1	1
	4	9.61	1	5.72	15.33	17.40	3	1
	5	13.62	2	4.30	17.92	18.13	4	1
	6	14.58	3	3.55	18.13	19.96	3	1
	7	15.62	3	5.93	21.55	22.74	3	2
	8	16.32	4	6.42	22.74	24.09	2	2
	9	20.48	3	5.54	26.02	27.62	2	1
	10	24.35	2	3.26	27.62	29.84	1	1
South	1	8.62	0	0.	8.62	8.96	0	1
	2	13.17	0	0.	13.17	16.25	0	1
	3	20.13	1	5.89	26.02	28.28	2	1
	4	21.65	2	6.62	28.28	28.54	1	1
	5	32.76	0	0.	32.76	38.27	0	1
	6	35.34	1	2.93	38.27	38.66	1	2
	7	38.44	1	3.34	41.78	42.83	2	1
	8	39.83	2	3.90	43.72	44.42	1	1
	9	47.14	1	.32	47.47	52.37	2	1
	10	47.15	2	5.22	52.37	52.92	3	1
East	1	7.08	1	10.32	17.40	17.92	5	1
	2	7.88	2	10.04	17.92	19.23	4	2
	3	8.79	3	11.16	19.96	21.55	3	1
	4	16.04	4	5.50	21.55	22.85	3	2
	5	16.66	5	6.19	22.85	23.57	2	2
	6	20.12	3	3.96	24.09	26.02	2	1
	7	22.94	2	6.90	29.84	30.70	3	1
	8	28.43	2	9.84	38.27	38.54	5	1
	9	28.69	3	9.85	38.54	41.78	4	1
	10	30.91	3	11.92	42.83	43.72	5	1

Table 12. SAMPLE SECTION OF SUM.REPORT OF MODEL 1.

AQn: average queue length(running) at each north unit's departure.
AWn: average waiting time(running) at each north unit's departure.
AQs and AWs are for south units, AQe and AWe are for east units.

No	AQn	AWn	AQs	AWs	AQe	AWe
1	0.	0.	0.	0.	1.76	10.32
2	.89	2.05	0.	0.	1.82	10.18
3	.61	1.81	.39	1.96	1.94	10.51
4	.89	2.79	.44	3.13	2.01	9.26
5	1.27	3.09	.38	2.50	2.01	8.64
6	1.29	3.17	.40	2.57	2.01	7.86
7	1.45	3.56	.50	2.68	1.90	7.73
8	1.48	3.92	.52	2.83	2.32	7.99
9	1.48	4.10	.49	2.56	2.33	8.20
10	1.45	4.01	.65	2.82	2.47	8.57
11	1.34	3.74	.66	3.04	2.53	8.75
12	.86	3.42	.67	2.97	3.05	9.95
13	.79	3.16	.85	3.20	3.47	11.42
14	.78	2.97	.91	3.43	3.60	12.85
15	.71	2.90	1.23	4.10	3.93	14.53
16	.72	2.93	1.43	5.04	3.94	15.93
17	.67	2.88	1.44	5.60	3.95	16.96
18	.62	2.72	1.52	6.36	3.99	17.54
19	.69	2.92	1.52	6.66	4.18	17.84
20	.73	3.09	1.55	6.69	4.20	18.04
21	.74	3.19	1.63	6.88	4.21	17.77
22	.72	3.04	1.64	6.96	4.51	17.83
23	.63	2.91	1.61	7.36	4.56	17.92
24	.58	2.79	1.60	7.16	4.63	18.20
25	.57	2.68	1.61	7.04	5.03	18.73
26	.55	2.65	1.60	6.94	5.11	19.13
27	.57	2.76	1.31	6.69	5.18	19.54
28	.61	2.84	1.27	6.47	5.40	20.12
29	.63	3.05	1.24	6.45	5.61	20.87
30	.63	3.03	1.54	6.96	5.72	21.29

Table 13. SAMPLE SECTION OF DIAGNOSTIC PRINTOUT OF MODEL 2.: Simulation conditions; $P_1 = 0.75$, mean service time = 2.5, TI(traffic intensity) = 0.5, interarrival and service time distribution = SS (see Table 8 and Figure 16).

NNU: new.north.unit, NSU: new.south.unit, NEU: new.east.unit
NS: north.service, SS: south.service, ES: east.service
U: a uniform random variable used for the next route selection.
Movements: movements at the intersection after the left event.
N,S,E: north, south, and east direction.
Queue: number of units in each queue after the left event.
Next route: routes selected for the first unit in each queue.

Event	Time	U	Movements (N, S, E)			Queue (N,S,E)			Next route (N,S,E)		
NNU	.39	.45	1	0	0	0	0	0	0	0	0
NNU	.51	.43	1	0	0	1	0	0	0	0	0
NS	4.62	.66	1	0	0	0	0	0	0	0	0
NEU	7.08	.52	1	0	0	0	0	1	0	0	1
NNU	7.51	.04	1	0	0	1	0	1	0	0	1
NEU	7.88	.44	1	0	0	1	0	2	0	0	1
NSU	8.62	.34	1	1	0	1	0	2	0	0	1
NEU	8.79	.54	1	1	0	1	0	3	0	0	1
NS	8.83	.42	0	1	1	1	0	2	1	0	0
SS	8.96	.23	0	0	1	1	0	2	1	0	0
NNU	9.61	.54	0	0	1	2	0	2	1	0	0
NSU	13.17	.07	0	1	1	2	0	2	1	0	0
NNU	13.62	.82	0	1	1	3	0	2	1	0	0
NNU	14.58	.33	0	1	1	4	0	2	1	0	0
ES	15.33	.26	1	1	0	3	0	2	0	0	1
NNU	15.62	.08	1	1	0	4	0	2	0	0	1
NEU	16.04	.73	1	1	0	4	0	3	0	0	1
SS	16.25	.50	1	0	0	4	0	3	0	0	1
NNU	16.32	.20	1	0	0	5	0	3	0	0	1
NEU	16.66	.57	1	0	0	5	0	4	0	0	1
NS	17.40	.44	0	0	1	5	0	3	1	0	0
ES	17.92	.91	1	0	2	4	0	2	0	0	0
NS	18.13	.31	1	0	2	3	0	2	0	0	0
ES	19.23	.55	1	0	0	3	0	2	0	0	1
NS	19.96	.91	2	0	1	2	0	1	0	0	0
NEU	20.12	.05	2	0	1	2	0	2	0	0	0
NSU	20.13	.37	2	1	1	2	0	2	0	0	0
NNU	20.48	.64	2	1	1	3	0	2	0	0	0
NS	21.14	.99	2	1	1	2	0	2	0	0	0
SS	21.43	.55	2	0	1	2	0	2	0	0	0
ES	21.55	.84	2	0	2	2	0	1	0	0	0
NSU	21.65	.83	2	0	2	2	1	1	0	2	0

Table 14. SAMPLE SECTION OF SINGLE.RUN.REPORT OF MODEL 2.

AT: arrival time of a unit.
Q1: queue length after an arrival.
Q2: queue length before departure of a unit.
WT: waiting time of each unit.
ST: departure time of a unit.
ET: time of service completion.
DIR: route number used by a unit.

	NO	AT	Q1	WT	ST	ET	Q2	DIR
North	1	.39	0	0.	.39	4.62	0	1
	2	.51	1	4.10	4.62	8.83	1	1
	3	7.51	1	7.82	15.33	17.40	4	1
	4	9.61	2	8.31	17.92	18.13	5	1
	5	13.62	3	4.51	18.13	19.96	4	1
	6	14.58	4	5.37	19.96	21.14	3	2
	7	15.62	4	5.52	21.14	22.49	3	2
	8	16.32	5	7.89	24.21	26.46	2	1
	9	20.48	3	8.21	28.69	28.95	2	1
	10	24.35	2	5.47	29.82	34.99	2	1
South	1	8.62	0	0.	8.62	8.96	0	1
	2	13.17	0	0.	13.17	16.25	0	1
	3	20.13	0	0.	20.13	21.43	0	1
	4	21.65	1	.84	22.49	24.09	1	2
	5	32.76	0	0.	32.76	38.27	0	1
	6	35.34	1	2.93	38.27	39.32	1	1
	7	38.44	1	.88	39.32	40.01	1	1
	8	39.83	1	.19	40.01	44.92	1	1
	9	47.14	0	0.	47.14	47.69	0	1
	10	47.15	1	.54	47.69	49.75	2	1
East	1	7.08	1	1.75	8.83	15.33	3	1
	2	7.88	2	9.52	17.40	17.92	4	1
	3	8.79	3	9.13	17.92	19.23	3	2
	4	16.04	3	3.91	19.96	21.55	2	1
	5	16.66	4	4.89	21.55	22.27	2	2
	6	20.12	2	2.15	22.27	24.21	1	1
	7	22.94	1	3.52	26.46	28.69	1	1
	8	28.43	1	.53	28.95	29.82	2	1
	9	28.69	2	6.29	34.99	35.26	4	1
	10	30.91	2	4.75	35.66	38.90	3	1

Table 15. SAMPLE SECTION OF SUM.REPORT OF MODEL 2.

AQn: average queue length(running) at each north unit's departure.
AWn: average waiting time(running) at each north unit's departure.
AQs and AWs are for south units, AQe and AWe are for east units.

No	AQn	AWn	AQs	AWs	AQe	AWe
1	0.	0.	0.	0.	.31	1.75
2	.89	2.05	0.	0.	1.26	5.63
3	1.31	3.97	0.	0.	1.31	6.80
4	1.77	5.06	.04	.21	1.38	6.08
5	1.80	4.95	.03	.17	1.42	5.84
6	1.91	5.02	.10	.63	1.41	5.22
7	1.95	5.09	.12	.66	1.32	4.98
8	1.95	5.44	.12	.60	1.23	4.42
9	1.95	5.75	.10	.54	1.34	4.63
10	1.92	5.72	.11	.54	1.38	4.64
11	1.78	5.70	.15	.68	1.43	4.68
12	1.31	5.23	.21	.97	1.42	4.70
13	1.22	5.00	.22	.99	1.39	4.56
14	1.21	5.02	.35	1.18	1.39	4.31
15	1.15	4.77	.54	1.63	1.28	4.02
16	1.15	4.58	.73	2.37	1.22	4.00
17	1.03	4.31	.74	2.73	1.18	3.91
18	.92	4.07	.79	3.19	1.08	3.73
19	.91	3.86	.79	3.23	.93	3.53
20	.88	3.78	.74	3.06	.90	3.39
21	.88	3.67	.72	2.94	.80	3.23
22	.83	3.69	.69	2.80	.78	3.08
23	.77	3.53	.68	2.68	.78	2.99
24	.72	3.53	.59	2.57	.90	3.25
25	.73	3.55	.58	2.47	.92	3.41
26	.74	3.55	.58	2.51	1.01	3.53
27	.74	3.52	.47	2.41	1.04	3.66
28	.77	3.61	.47	2.39	1.11	3.97
29	.77	3.73	.46	2.31	1.11	4.29
30	.98	4.23	.43	2.23	1.12	4.22

Table 16. SIMULATION OUTPUTS FOR VARIOUS CONDITIONS OF MODEL 1

P_1: probability taking route1.
MS: mean service time.
TI: traffic intensity(see Table 8).
A.S: interarrival and service time distribution (see Figure 16).
WT: mean waiting time (see equation (3.7) in chapter III.)
SD: std.dev. of WT(see equation (3.8) in chapter III.)

P_1	MS	TI	A.S	NORTH WT	NORTH SD	SOUTH WT	SOUTH SD	EAST WT	EAST SD
0.75	2.5	0.5	SS	16.71	1.63	28.40	2.39	155.93	18.38
			SR	11.38	.77	22.67	1.82	120.57	10.39
			RS	10.90	.65	26.29	3.21	159.92	16.34
			RR	4.87	.29	13.31	.96	99.75	12.45
		0.4	SS	7.27	.39	9.72	.75	16.41	1.35
			SR	4.93	.32	6.23	.37	11.78	1.36
			RS	4.72	.33	6.90	.38	11.50	.66
			RR	2.09	.08	3.18	.15	4.53	.18
		0.33	SS	4.49	.20	5.66	.22	8.54	.67
			SR	2.68	.10	3.30	.10	4.41	.19
			RS	2.45	.12	2.98	.10	4.09	.18
			RR	1.15	.04	1.49	.05	1.92	.04
	5.0	0.5	SS	37.56	2.83	82.31	13.74	337.10	27.53
			SR	23.54	1.47	50.80	5.42	258.47	24.42
			RS	22.59	1.94	50.35	5.91	344.53	37.02
			RR	10.40	.49	30.51	2.63	163.22	20.72
		0.4	SS	14.46	.96	16.68	.81	32.62	3.86
			SR	9.11	.38	12.22	.53	22.31	1.47
			RS	9.00	.46	12.77	1.07	22.56	1.79
			RR	3.90	.11	5.91	.27	7.99	.30
		0.33	SS	9.84	.49	10.49	.43	16.71	1.17
			SR	5.58	.17	6.27	.24	8.77	.36
			RS	5.39	.25	6.54	.35	8.86	.46
			RR	2.43	.08	3.00	.14	3.83	.15

Table 17. CONTINUED FROM THE PREVIOUS TABLE.

P_1	MS	TI	A.S	NORTH WT	NORTH SD	SOUTH WT	SOUTH SD	EAST WT	EAST SD
0.50	2.5	0.5	SS	21.13	2.47	67.87	12.09	17.75	1.30
			SR	11.42	1.06	29.23	3.71	9.33	.66
			RS	14.80	1.57	47.61	8.42	11.88	1.64
			RR	5.40	.33	16.00	1.97	4.50	.20
		0.4	SS	6.85	.30	10.05	1.18	7.19	.43
			SR	4.49	.20	6.41	.47	4.43	.21
			RS	4.51	.21	7.31	.41	4.14	.25
			RR	1.87	.06	3.20	.18	1.73	.06
		0.33	SS	4.37	.19	4.99	.27	4.16	.15
			SR	2.59	.11	3.04	.16	2.57	.13
			RS	2.64	.13	3.34	.17	2.38	.13
			RR	1.11	.03	1.27	.04	.98	.03
	5.0	0.5	SS	36.04	3.69	106.57	22.53	39.94	4.55
			SR	21.93	1.64	59.29	6.06	20.82	1.53
			RS	25.03	2.11	79.50	12.27	26.06	1.86
			RR	11.87	.89	32.57	3.47	9.93	.64
		0.4	SS	15.11	1.41	21.31	1.41	14.39	.82
			SR	8.30	.30	12.25	.83	8.20	.57
			RS	9.03	.44	14.88	1.01	8.50	.39
			RR	3.94	.12	6.04	.29	3.59	.14
		0.33	SS	8.87	.62	10.31	.47	9.07	.45
			SR	5.44	.21	6.12	.34	5.06	.17
			RS	5.15	.23	6.50	.25	4.61	.21
			RR	2.24	.07	2.91	.10	2.06	.06

Table 18. CONTINUED FROM THE PREVIOUS TABLE.

P_1	MS	TI	A.S	NORTH WT	NORTH SD	SOUTH WT	SOUTH SD	EAST WT	EAST SD
0.25	2.5	0.5	SS	16.77	1.21	27.61	2.99	6.15	.39
			SR	12.16	1.34	22.73	3.41	3.48	.16
			RS	13.12	.98	28.20	5.13	3.44	.22
			RR	5.79	.35	11.00	1.36	1.37	.05
		0.4	SS	7.44	.35	8.53	.62	3.29	.16
			SR	4.58	.18	4.85	.25	2.06	.09
			RS	4.36	.19	5.20	.25	1.84	.07
			RR	1.95	.06	2.12	.11	.76	.04
		0.33	SS	4.53	.28	4.61	.19	2.43	.09
			SR	2.53	.07	2.71	.10	1.47	.07
			RS	2.49	.10	2.87	.16	1.23	.07
			RR	1.06	.03	1.14	.05	.48	.02
	5.0	0.5	SS	38.06	3.29	76.42	11.18	11.85	.80
			SR	21.42	1.34	34.54	2.93	7.16	.38
			RS	25.02	2.25	50.95	6.00	7.01	.39
			RR	10.98	.61	17.03	1.16	2.58	.11
		0.4	SS	14.03	.82	17.44	1.41	6.97	.40
			SR	8.47	.28	8.72	.37	4.42	.30
			RS	8.50	.51	10.00	.47	4.04	.23
			RR	3.90	.16	4.35	.21	1.48	.04
		0.33	SS	8.91	.36	8.26	.25	4.71	.21
			SR	5.29	.17	5.67	.21	2.94	.10
			RS	4.72	.13	5.22	.19	2.26	.08
			RR	2.24	.08	2.34	.06	1.00	.03

Table 19. SIMULATION OUTPUTS FOR VARIOUS CONDITIONS OF MODEL 2

P_1: probability taking route1.
MS: mean service time.
TI: traffic intensity(see Table 8).
A.S: interarrival and service time distribution (see Figure 16).
WT: mean waiting time (see equation (3.7) in chapter III.)
SD: std.dev. of WT(see equation (3.8) in chapter III.)

P_1	MS	TI	A.S	NORTH WT	NORTH SD	SOUTH WT	SOUTH SD	EAST WT	EAST SD
0.75	2.5	0.5	SS	9.09	.82	4.10	.21	11.89	1.23
			SR	5.22	.27	2.61	.11	6.66	.44
			RS	6.14	.33	2.13	.08	7.19	.46
			RR	2.36	.13	.84	.03	3.00	.18
		0.4	SS	4.65	.26	2.75	.16	5.32	.31
			SR	2.87	.12	1.56	.05	2.96	.13
			RS	2.57	.10	1.26	.06	2.95	.12
			RR	1.07	.04	.51	.02	1.19	.04
		0.33	SS	3.03	.16	1.69	.07	3.30	.14
			SR	1.87	.07	1.11	.04	2.11	.12
			RS	1.51	.06	.80	.04	1.73	.09
			RR	.63	.02	.27	.01	.73	.02
	5.0	0.5	SS	19.81	1.66	8.38	.35	26.59	1.92
			SR	11.62	.71	5.48	.48	15.15	1.44
			RS	13.17	.81	4.74	.27	16.95	2.65
			RR	4.75	.24	1.68	.06	5.99	.32
		0.4	SS	8.76	.48	5.18	.19	10.18	.55
			SR	5.67	.27	3.28	.09	6.39	.26
			RS	5.10	.23	2.31	.09	5.85	.35
			RR	2.08	.07	.84	.03	2.40	.08
		0.33	SS	5.71	.29	3.63	.17	6.53	.24
			SR	3.91	.20	2.24	.07	4.15	.21
			RS	3.18	.13	1.52	.05	3.44	.13
			RR	1.27	.05	.57	.02	1.50	.04

Table 20. CONTINUED FROM THE PREVIOUS TABLE.

P_1	MS	TI	A.S	NORTH WT	NORTH SD	SOUTH WT	SOUTH SD	EAST WT	EAST SD
0.50	2.5	0.5	SS	6.42	.46	6.29	.34	5.66	.24
			SR	3.95	.19	4.03	.20	3.50	.09
			RS	3.61	.19	3.79	.20	3.96	.19
			RR	1.40	.05	1.44	.05	1.39	.04
		0.4	SS	3.60	.19	3.89	.19	3.60	.14
			SR	2.22	.07	2.08	.07	2.09	.08
			RS	1.96	.09	1.83	.08	1.95	.12
			RR	.74	.02	.74	.02	.74	.02
		0.33	SS	2.35	.11	2.28	.10	2.40	.09
			SR	1.49	.04	1.49	.06	1.43	.05
			RS	1.15	.04	1.18	.04	1.18	.07
			RR	.46	.01	.49	.02	.46	.01
	5.0	0.5	SS	13.03	.77	14.24	1.06	14.05	1.13
			SR	8.25	.48	7.54	.26	7.49	.36
			RS	7.94	.49	7.88	.47	8.61	.51
			RR	2.91	.13	2.89	.13	2.81	.12
		0.4	SS	7.27	.28	6.80	.30	6.59	.21
			SR	4.20	.18	4.45	.16	4.58	.15
			RS	3.39	.11	3.45	.13	3.73	.13
			RR	1.47	.05	1.42	.03	1.45	.05
		0.33	SS	4.58	.18	4.43	.17	5.19	.23
			SR	2.96	.12	2.86	.07	3.07	.12
			RS	2.33	.08	2.39	.10	2.45	.11
			RR	1.01	.04	.96	.03	.98	.03

Table 21. CONTINUED FROM THE PREVIOUS TABLE.

P_1	MS	TI	A.S	NORTH WT	NORTH SD	SOUTH WT	SOUTH SD	EAST WT	EAST SD
0.25	2.5	0.5	SS	11.13	.84	12.27	.66	4.51	.24
			SR	6.38	.63	7.24	.54	2.59	.10
			RS	5.64	.36	6.69	.46	2.22	.13
			RR	2.31	.07	2.84	.10	.80	.03
		0.4	SS	4.65	.28	4.81	.27	2.24	.13
			SR	2.65	.09	2.81	.11	1.67	.08
			RS	2.62	.12	3.01	.16	1.20	.05
			RR	1.01	.02	1.12	.02	.45	.02
		0.33	SS	2.85	.15	3.14	.11	1.75	.09
			SR	1.87	.09	1.91	.06	1.16	.03
			RS	1.60	.04	1.85	.08	.76	.04
			RR	.61	.02	.75	.02	.28	.01
	5.0	0.5	SS	18.62	2.82	20.37	1.70	7.83	.42
			SR	12.13	1.09	13.01	.89	5.27	.30
			RS	12.60	.78	13.08	.60	4.87	.24
			RR	4.86	.26	6.16	.26	1.80	.08
		0.4	SS	8.54	.41	9.02	.37	5.71	.26
			SR	5.76	.25	6.13	.25	3.07	.10
			RS	4.98	.15	5.24	.21	2.37	.12
			RR	2.18	.05	2.46	.08	.91	.03
		0.33	SS	6.09	.26	6.29	.18	3.64	.15
			SR	3.62	.13	4.03	.15	2.23	.07
			RS	2.99	.08	3.41	.10	1.48	.06
			RR	1.27	.05	1.47	.03	.57	.02

APPENDIX B. A PROGRAM LISTING OF MODEL 1.

```
PREAMBLE    ''an intersection simulation
 normally, mode is undefined
 event notices include north.unit.generator,
                       south.unit.generator,
                       east.unit.generator,

                       new.north.unit,
                       new.south.unit,
                       new.east.unit,

                       north.service,
                       south.service,
                       east.service

 every north.service has an north.pointer
 every south.service has an south.pointer
 every east.service has an east.pointer

 define north.pointer,
        south.pointer,
        east.pointer as pointer variables

 temporary entities
 every unit may belong to the north.queue
            may belong to the south.queue
            may belong to the east.queue

 the system owns the north.queue ,
                 the south.queue,
                 the east.queue

 define north.queue,
        south.queue
        and east.queue as fifo sets

 define north.run,
        south.run,
        east.run,
        next.north.route,
        next.south.route,
        next.east.route,
        N1,
        replication,NR,
        in,jn,kn,is,js,ks,ie,je,ke,
        saveseed1,saveseed2,saveseed3 as integer variables

 define AQ.north.n1, AQ.south.n1, AQ.east.n1,
        AW.north.n1, AW.south.n1, AW.east.n1,
        alpha1, beta1, alpha2, beta2,
        P1, current.time,
        s1,s2,s3,r1,r2,r3 as real variables
```

```
define north.dir,south.dir,east.dir
       as 1-dimensional integer array

define arrival.time.of.north.unit,
       departure.time.of.north.unit,
       end.time.of.north.service,
       waiting.time.of.north.unit,
       north.queue.length1,
       north.queue.length2,

       arrival.time.of.south.unit,
       departure.time.of.south.unit,
       end.time.of.south.service,
       waiting.time.of.south.unit,
       south.queue.length1,
       south.queue.length2,

       arrival.time.of.east.unit,
       departure.time.of.east.unit,
       end.time.of.east.service,
       waiting.time.of.east.unit,
       east.queue.length1,
       and east.queue.length2,
       AQ.of.north.unit, AQ.of.south.unit, AQ.of.east.unit,
       AW.of.north.unit, AW.of.south.unit, AW.of.east.unit
       as 1-dimensional real array

tally ave.que.north.n1 as the mean and
      sd.ave.que.north.n1 as the std.dev of AQ.north.n1
tally ave.wt.north.n1 as the mean and
      sd.ave.wt.north.n1 as the std.dev of AW.north.n1

tally ave.que.south.n1 as the mean and
      sd.ave.que.south.n1 as the std.dev of AQ.south.n1
tally ave.wt.south.n1 as the mean and
      sd.ave.wt.south.n1 as the std.dev of AW.south.n1

tally ave.que.east.n1 as the mean and
      sd.ave.que.east.n1 as the std.dev of AQ.east.n1
tally ave.wt.east.n1 as the mean and
      sd.ave.wt.east.n1 as the std.dev of AW.east.n1

accumulate AQ.north.unit as the mean of n.north.queue
accumulate AQ.south.unit as the mean of n.south.queue
accumulate AQ.east.unit as the mean of n.east.queue

priority order is north.unit.generator,
                  south.unit.generator,
                  east.unit.generator

END ''preamble


MAIN '' program starts execution here.
 define s.que.north,
```

```
        s.wt.north,
        s.que.south,
        s.wt.south,
        s.que.east,
        s.wt.east,
        inverse,traffic.intensity as real variables

call read.data
     saveseed1 = seed.v(s1)
     saveseed2 = seed.v(s2)
     saveseed3 = seed.v(s3)


for P1 = 0.25 to 0.75 by 0.25
        do
also for alpha2 = 2.5 to 5.0 by 2.5
        do
also for inverse = 2 to 3 by .5
        do
        traffic.intensity = 1/inverse
        alpha1 = alpha2*inverse
        r1 = 1
        r2 = 1
        r3 = 1
also for beta1 = 1 to 4 by 3
        do
also for beta2 = 1 to 4 by 3
        do
        print 2 line thus
        ==================================
         P1    a1    b1    a2    b2   TI
        print 2 line with P1,alpha1,beta1,alpha2,
                       beta2,traffic.intensity thus
        *.** **.** **.** **.** **.** *.**
        ==================================
for replication = 1 to NR
     do
     let seed.v(s1) = saveseed1
     let seed.v(s2) = saveseed2
     let seed.v(s3) = saveseed3

          call initialize
          start simulation
          call single.run.report
          call sum.report

     AQ.north.n1 = AQ.of.north.unit(N1)
     AQ.south.n1 = AQ.of.south.unit(N1)
     AQ.east.n1  = AQ.of.east.unit(N1)
     AW.north.n1 = AW.of.north.unit(N1)
     AW.south.n1 = AW.of.south.unit(N1)
     AW.east.n1  = AW.of.east.unit(N1)

     let time.v = 0
     let saveseed1 = seed.v(s1)
     let saveseed2 = seed.v(s2)
```

```
    let saveseed3 = seed.v(s3)

    let in = 0
        is = 0
        ie = 0

        jn = 0
        js = 0
        je = 0

        kn = 0
        ks = 0
        ke = 0

if replication = NR and NR > 1
        s.que.north = sd.ave.que.north.n1/sqrt.f(NR - 1)
        s.que.south = sd.ave.que.south.n1/sqrt.f(NR - 1)
        s.que.east  = sd.ave.que.east.n1/sqrt.f(NR - 1)

        s.wt.north = sd.ave.wt.north.n1/sqrt.f(NR - 1)
        s.wt.south = sd.ave.wt.south.n1/sqrt.f(NR - 1)
        s.wt.east  = sd.ave.wt.east.n1/sqrt.f(NR - 1)


      ''print estimators of mean queue length (running)
      ''at the N1th unit's departure and mean waiting time
      ''(running) of the N1th unit and std.dev of each.

        print 5 lines with ave.que.north.n1,
                           s.que.north,
                           ave.que.south.n1,
                           s.que.south,
                           ave.que.east.n1,
                           s.que.east,
                           ave.wt.north.n1,s.wt.north,
                           ave.wt.south.n1,s.wt.south,
                           ave.wt.east.n1,s.wt.east thus
 ==================== Main ====================
    See main routine for information of data.
 ==============================================
 **.***  **.***  **.***  **.***  **.***  **.***
 **.***  **.***  **.***  **.***  **.***  **.***
always

release arrival.time.of.north.unit,
        departure.time.of.north.unit,
        end.time.of.north.service,
        waiting.time.of.north.unit,
        north.queue.length2,
        arrival.time.of.south.unit,
        departure.time.of.south.unit,
        end.time.of.south.service,
        waiting.time.of.south.unit,
        south.queue.length2,
        arrival.time.of.east.unit,
        departure.time.of.east.unit,
```

```
          end.time.of.east.service,
          waiting.time.of.east.unit,
          east.queue.length2,
          AQ.of.north.unit,
          AQ.of.south.unit,
          AQ.of.east.unit,
          AW.of.north.unit,
          AW.of.south.unit,
          AW.of.east.unit,
          north.queue.length1,
          south.queue.length1,
          east.queue.length1,
          north.dir,south.dir,east.dir

 loop

 reset  the totals of AQ.north.N1,AQ.south.N1,AQ.east.N1,
                      AW.north.N1,AW.south.N1,AW.east.N1
 loop
END ''MAIN


ROUTINE to READ.DATA
 print 4 lines thus
 ============================================
    See MAIN routine to see other factors
         that are not asked here !
 ============================================
 print 2 lines thus
       enter the seed stream number1,2,3
       s1, s2 ,s3             (1 - 10)
 read s1,s2,s3

 print 3 lines thus
       enter the desired number of units
            and replications simulated !
        N1, NR
 read N1,NR
END '' READ.DATA


ROUTINE to INITIALIZE

 let seed.v(s1) = saveseed1
 let seed.v(s2) = saveseed2
 let seed.v(s3) = saveseed3

 schedule a north.unit.generator in
            gamma.f(r1*alpha1,beta1,s1) minutes
 schedule a south.unit.generator in
            gamma.f(r2*alpha1,beta1,s1) minutes
 schedule a east.unit.generator in
            gamma.f(r3*alpha1,beta1,s1) minutes
 print 4 lines thus
 ================= 6 event routine =================
                      Movements    Queue   Next route
```

Event	Time	U	(N, S, E)	(N,S,E)	(N,S,E)

```
 reserve arrival.time.of.north.unit,
         departure.time.of.north.unit,
         end.time.of.north.service,
         waiting.time.of.north.unit
         and north.queue.length2 as 2000

 reserve north.queue.length1,
         south.queue.length1,
         east.queue.length1 as 2000

 reserve arrival.time.of.south.unit,
         departure.time.of.south.unit,
         end.time.of.south.service,
         waiting.time.of.south.unit
         and south.queue.length2 as 2000

 reserve arrival.time.of.east.unit,
         departure.time.of.east.unit,
         end.time.of.east.service,
         waiting.time.of.east.unit
         and east.queue.length2 as 2000

 reserve north.dir,south.dir,east.dir as 2000

 reserve AQ.of.north.unit,AQ.of.south.unit,AQ.of.east.unit,
         AW.of.north.unit,AW.of.south.unit,AW.of.east.unit as 2000
END  ''INITIALIZE


EVENT NORTH.UNIT.GENERATOR saving the event notice
 if jn < N1 or js < N1 or je < N1
       schedule a new.north.unit now
       reschedule this north.unit.generator in
                        gamma.f(r1*alpha1, beta1,s1) minutes
 always
END ''NORTH.UNIT.GENERATOR


EVENT SOUTH.UNIT.GENERATOR saving the event notice
 if jn < N1 or js < N1 or je < N1
       schedule a new.south.unit now
       reschedule this south.unit.generator in
                  gamma.f(r1*alpha1, beta1,s1) minutes
 always
END ''SOUTH.UNIT.GENERATOR


EVENT EAST.UNIT.GENERATOR saving the event notice
 if jn < N1 or js < N1 or je < N1
       schedule a new.east.unit now
       reschedule this east.unit.generator in
                  gamma.f(r3*alpha1,beta1,s1) minutes
 always
END ''EAST.UNIT.GENERATOR
```

```
EVENT NEW.NORTH.UNIT
 define U as an real variable
 create an unit
 let  in = in + 1
 current.time = time.v*hours.v*minutes.v
 arrival.time.of.north.unit(in) = current.time
 U = uniform.f(0,1,s3)

 if north.run <> 0 or n.north.queue > 0
 go to 'case3'
        else
        if U < P1
             next.north.route = 1
             go to 'case1'
        else
             next.north.route = 2
             go to 'case2'

'case1'
 if east.run <> 1
       north.run = 1
       next.north.route = 0
       jn = jn + 1
       schedule an north.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.north.unit(jn) = current.time
       AQ.of.north.unit(jn) = AQ.north.unit
 else
'case3'
       file this unit in the north.queue
       north.queue.length2(in) = n.north.queue
 always
 go to 'end'

'case2'
 if south.run = 0 and east.run <> 1
       let north.run = 2
       jn = jn + 1
       next.north.route = 0
       schedule an north.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.north.unit(jn) = current.time
       AQ.of.north.unit(jn) = AQ.north.unit
 else
       go to 'case3'
 always

'end'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
  NNU   **.**   .**   ** ** **   ** ** **   ** ** **

END   ''NEW.NORTH.UNIT
```

```
EVENT NEW.SOUTH.UNIT
 define U as real variable
 create an unit
 is = is + 1
 current.time = time.v*hours.v*minutes.v
 arrival.time.of.south.unit(is) = current.time
 let U = uniform.f(0,1,s3)

 if n.south.queue <> 0 or south.run <> 0
 go to 'case3'
        else
        if U < P1
           next.south.route = 1
           go to 'case1'
        else
           next.south.route = 2
           go to 'case2'

'case1'
 if north.run <> 2 and east.run = 0
       south.run = 1
       next.south.route = 0
       js = js + 1

       schedule an south.service giving unit
                in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.south.unit(js) = current.time
       AQ.of.south.unit(js) = AQ.south.unit
 else

'case3'
       file this unit in the south.queue
       south.queue.length2(is) = n.south.queue
 always
 go to 'end'

'case2'
 if north.run <> 2
       let south.run = 2
       js = js + 1
       next.south.route = 0
       schedule an south.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.south.unit(js) = current.time
       AQ.of.south.unit(js) = AQ.south.unit
 else
       go to 'case3'
 always

'end'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
  NSU   **.**   .**   ** ** **   ** ** **   ** ** **
```

```
END ''NEW.SOUTH.UNIT


EVENT NEW.EAST.UNIT
 define U as real variable
 create an unit
 ie = ie + 1
 current.time = time.v*hours.v*minutes.v
 arrival.time.of.east.unit(ie) = current.time
 U = uniform.f(0,1,s3)

 if n.east.queue > 0 or east.run <> 0
 go to 'case3'
        else
        if U < P1
             next.east.route = 1
             go to 'case1'
        else
            next.east.route = 2
            go to 'case2'
'case1'
 if south.run <> 1 and north.run = 0
       east.run = 1
       next.east.route = 0
       je = je + 1
       schedule an east.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.east.unit(je) = current.time
       AQ.of.east.unit(je) = AQ.east.unit
 else
'case3'
       file this unit in the east.queue
       east.queue.length2(ie) = n.east.queue
 always
 go to 'end'

'case2'
 if south.run <> 1
       east.run = 2
       je = je + 1
       next.east.route = 0
       schedule an east.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.east.unit(je) = current.time
       AQ.of.east.unit(je) = AQ.east.unit
  else
       go to 'case3'
  always

'end'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
```

```
  NEU    **.**    .**    ** ** **    ** ** **    ** ** **
END    ''NEW.EAST.UNIT


EVENT NORTH.SERVICE given unit
 define unit as a pointer variable
 define U as real variable
 destroy this unit
 kn = kn + 1
 current.time = time.v*hours.v*minutes.v
 end.time.of.north.service(kn) = current.time
 U = uniform.f(0,1,s3)

 if n.north.queue > 0 and U < P1
       next.north.route = 1
 always

 if n.north.queue > 0 and U >= P1
       next.north.route = 2
 always

 if north.run = 2
       go to 'service2'
 else
       north.dir(kn) = 1
       north.run = 0

 if south.run <> 1 and next.east.route = 1
       east.run = 1
       je = je + 1
       departure.time.of.east.unit(je) = current.time
       waiting.time.of.east.unit(je) =
          current.time - arrival.time.of.east.unit(je)
       east.queue.length1(je) = n.east.queue
       remove the first unit from the east.queue
       AQ.of.east.unit(je) = AQ.east.unit
       next.east.route = 0
       schedule an east.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always

'next'
 if east.run <> 1 and next.north.route = 1
       jn = jn + 1
       north.run = 1
       departure.time.of.north.unit(jn) = current.time
       waiting.time.of.north.unit(jn)   =
          current.time - arrival.time.of.north.unit(jn)
       north.queue.length1(jn) = n.north.queue
       remove the first unit from the north.queue
       AQ.of.north.unit(jn) = AQ.north.unit
       next.north.route = 0
       schedule an north.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always
```

```
if east.run <> 1 and next.north.route = 2 and south.run = 0
      jn = jn + 1
      north.run = 2
      departure.time.of.north.unit(jn) = current.time
      waiting.time.of.north.unit(jn)    =
         current.time - arrival.time.of.north.unit(jn)
      north.queue.length1(jn)  = n.north.queue
      remove the first unit from the north.queue
      AQ.of.north.unit(jn) = AQ.north.unit
      next.north.route = 0
      schedule an north.service giving unit
                  in gamma.f(alpha2,beta2,s2) minutes
always
go to 'print'

'service2'
north.dir(kn) = 2
north.run     = 0

if next.east.route = 1
      je = je + 1
      east.run = 1
      departure.time.of.east.unit(je) = current.time
      waiting.time.of.east.unit(je) =
         current.time - arrival.time.of.east.unit(je)
      east.queue.length1(je) = n.east.queue
      remove the first unit from the east.queue
      AQ.of.east.unit(je) = AQ.east.unit
      next.east.route = 0
      schedule an east.service giving unit
                  in gamma.f(alpha2,beta2,s2) minutes
always

if east.run = 0 and next.south.route = 1
      js = js + 1
      south.run = 1
      departure.time.of.south.unit(js) = current.time
      waiting.time.of.south.unit(js) = current.time -
                       arrival.time.of.south.unit(js)
      south.queue.length1(js) = n.south.queue
      remove the first unit from the south.queue
      AQ.of.south.unit(js) = AQ.south.unit
      next.south.route = 0
      schedule an south.service giving unit
               in gamma.f(alpha2,beta2,s2) minutes
always
if next.south.route = 2
      js = js + 1
      south.run = 2
      departure.time.of.south.unit(js) = current.time
      waiting.time.of.south.unit(js) =
         current.time - arrival.time.of.south.unit(js)
      south.queue.length1(js) = n.south.queue
      remove the first unit from the south.queue
      AQ.of.south.unit(js) = AQ.south.unit
      next.south.route = 0
```

```
      schedule an south.service giving unit
               in gamma.f(alpha2,beta2,s2) minutes
 always
 go to 'next'

'print'
 print 1 line with current.time,U,
                  north.run,south.run,east.run,
                  n.north.queue,n.south.queue,n.east.queue,
                  next.north.route,next.south.route,
                  next.east.route thus
  NS    **.**   .**   ** ** **   ** ** **   ** ** **
END  ''NORTH.SERVICE


EVENT SOUTH.SERVICE given unit
 define U as real variable
 define unit as a pointer variable
 ks = ks + 1
 destroy this unit
 current.time = time.v*hours.v*minutes.v
 end.time.of.south.service(ks) = current.time
 U = uniform.f(0,1,s3)

 if n.south.queue > 0 and U < P1
       next.south.route = 1
 always

 if n.south.queue > 0 and U >= P1
       next.south.route = 2
 always

 if south.run = 2
       go to 'service2'
 else
       south.dir(ks) = 1
       south.run     = 0

 if next.east.route = 1 and north.run = 0
       je = je + 1
       east.run = 1
       departure.time.of.east.unit(je) = current.time
       waiting.time.of.east.unit(je) =
          current.time - arrival.time.of.east.unit(je)
       east.queue.length1(je) = n.east.queue
       remove the first unit from the east.queue
       AQ.of.east.unit(je) = AQ.east.unit
       next.east.route = 0
       schedule an east.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
 always

 if next.east.route = 2
       je = je + 1
       east.run = 2
       departure.time.of.east.unit(je) = current.time
```

```
       waiting.time.of.east.unit(je) = current.time -
                              arrival.time.of.east.unit(je)
       east.queue.length1(je) = n.east.queue
       remove the first unit from the east.queue
       AQ.of.east.unit(je) = AQ.east.unit
       next.east.route = 0
       schedule an east.service giving unit
                in gamma.f(alpha2,beta2,s2) minutes
 always
'next'
 if next.north.route = 2  and east.run <> 1
       jn = jn + 1
       north.run = 2
       departure.time.of.north.unit(jn) = current.time
       waiting.time.of.north.unit(jn) =
          current.time - arrival.time.of.north.unit(jn)
       north.queue.length1(jn) = n.north.queue
       remove the first unit from the north.queue
       AQ.of.north.unit(jn) = AQ.north.unit
       next.north.route = 0
       schedule an north.service giving unit
                in gamma.f(alpha2,beta2,s2) minutes
 always

 if east.run = 0 and north.run <> 2 and next.south.route = 1
       js = js + 1
       south.run = 1
       departure.time.of.south.unit(js) = current.time
       waiting.time.of.south.unit(js) =
          current.time - arrival.time.of.south.unit(js)
       south.queue.length1(js) = n.south.queue
       remove the first unit from the south.queue
       AQ.of.south.unit(js) = AQ.south.unit
       next.south.route = 0
       schedule an south.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
 always

 if north.run <> 2 and next.south.route = 2
       js = js + 1
       south.run = 2
       departure.time.of.south.unit(js) = current.time
       waiting.time.of.south.unit(js) =
          current.time - arrival.time.of.south.unit(js)
       south.queue.length1(js) = n.south.queue
       remove the first unit from the south.queue
       AQ.of.south.unit(js) = AQ.south.unit
       next.south.route = 0
       schedule an south.service giving unit
                in gamma.f(alpha2,beta2,s2) minutes
 always
 go to 'print'

'service2'
 south.dir(ks) = 2
 south.run     = 0
```

```
 go to 'next'

'print'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
  SS    **.**    .**    ** ** **    ** ** **    ** ** **
END ''SOUTH.SERVICE


EVENT EAST.SERVICE given unit
 define U as real variable
 define unit as a pointer variable
 destroy this unit
 ke = ke + 1
 current.time = time.v*hours.v*minutes.v
 end.time.of.east.service(ke) = current.time
 U = uniform.f(0,1,s3)

 if the east.queue is not empty  and U < P1
        next.east.route = 1
 always

 if the east.queue is not empty and U >= P1
        next.east.route = 2
 always

 if east.run = 2
        go to 'service2'
 else
        east.dir(ke) = 1
        east.run     = 0

 if next.north.route = 1
        jn = jn + 1
        north.run = 1
        departure.time.of.north.unit(jn) = current.time
        waiting.time.of.north.unit(jn) =
           current.time - arrival.time.of.north.unit(jn)
        north.queue.length1(jn) = n.north.queue
        remove the first unit from the north.queue
        AQ.of.north.unit(jn) = AQ.north.unit
        next.north.route = 0
        schedule an north.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always

 if next.north.route = 2 and south.run <> 2
        jn = jn + 1
        north.run = 2
        departure.time.of.north.unit(jn) = current.time
        waiting.time.of.north.unit(jn) =
           current.time - arrival.time.of.north.unit(jn)
        north.queue.length1(jn) = n.north.queue
```

```
       remove the first unit from the north.queue
       AQ.of.north.unit(jn) = AQ.north.unit
       next.north.route = 0
       schedule an north.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
always

'next'
if next.south.route = 1 and north.run <> 2
       js = js + 1
       south.run = 1
       departure.time.of.south.unit(js) = current.time
       waiting.time.of.south.unit(js) =
          current.time - arrival.time.of.south.unit(js)
       south.queue.length1(js) = n.south.queue
       remove the first unit from the south.queue
       AQ.of.south.unit(js) = AQ.south.unit
       next.south.route = 0
       schedule an south.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
always

if north.run = 0 and south.run <> 1 and next.east.route = 1
       je = je + 1
       east.run =1
       departure.time.of.east.unit(je) = current.time
       waiting.time.of.east.unit(je) =
          current.time - arrival.time.of.east.unit(je)
       east.queue.length1(je) = n.east.queue
       remove the first unit from the east.queue
       AQ.of.east.unit(je) = AQ.east.unit
       next.east.route = 0
       schedule an east.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
always

if south.run <> 1 and next.east.route = 2
       je = je + 1
       east.run = 2
       departure.time.of.east.unit(je) = current.time
       waiting.time.of.east.unit(je) =
          current.time - arrival.time.of.east.unit(je)
       east.queue.length1(je) = n.east.queue
       remove the first unit from the east.queue
       AQ.of.east.unit(je) = AQ.east.unit
       next.east.route = 0
       schedule an east.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
always
go to 'print'

'service2'
east.dir(ke) = 2
east.run = 0

go to 'next'
```

```
'print'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
  ES    **.**    .**    ** ** **    ** ** **    ** ** **

END ''EAST.SERVICE


ROUTINE to SINGLE.RUN.REPORT
 define nn,ns,ne as integer variables
 print 3 line thus
 ============== Single.run.report ===============
 NO     AT      Q1    WT       ST        ET      Q2   DIR
 ===============================================
 for nn = 1 to in
   do
   print 1 line with nn,
   arrival.time.of.north.unit(nn),
   north.queue.length2(nn),
   waiting.time.of.north.unit(nn),
   departure.time.of.north.unit(nn),
   end.time.of.north.service(nn),
   north.queue.length1(nn),north.dir(nn) thus
***    **.**    **    **.**    **.**    **.**    **    **
loop

for ns = 1 to is
   do
   print 1 line with ns,
   arrival.time.of.south.unit(ns),
   south.queue.length2(ns),
   waiting.time.of.south.unit(ns),
   departure.time.of.south.unit(ns),
   end.time.of.south.service(ns),
   south.queue.length1(ns),south.dir(ns) thus
***    **.**    **    **.**    **.**    **.**    **    **
loop

for ne = 1 to ie
   do
   print 1 line with ne,
   arrival.time.of.east.unit(ne),
   east.queue.length2(ne),
   waiting.time.of.east.unit(ne),
   departure.time.of.east.unit(ne),
   end.time.of.east.service(ne),
   east.queue.length1(ne),east.dir(ne) thus
***    **.**    **    **.**    **.**    **.**    **    **
loop
END ''SINGLE.RUN.REPORT
```

```
ROUTINE to SUM.REPORT
 define ii as integer variables
 define sum.of.north.wt,
        sum.of.south.wt,
        sum.of.east.wt as 1-dimensional real array
 define sum.north.wt,
        sum.south.wt,
        sum.east.wt as real variables
 reserve sum.of.north.wt,
         sum.of.south.wt,
         sum.of.east.wt as 2000

 for ii = 1 to N1
 do
      sum.north.wt = sum.north.wt +
                     waiting.time.of.north.unit(ii)
      sum.of.north.wt(ii) = sum.north.wt
      AW.of.north.unit(ii) = sum.of.north.wt(ii)/ii

      sum.south.wt = sum.south.wt +
                     waiting.time.of.south.unit(ii)
      sum.of.south.wt(ii) = sum.south.wt
      AW.of.south.unit(ii) = sum.of.south.wt(ii)/ii

      sum.east.wt = sum.east.wt +
                    waiting.time.of.east.unit(ii)
      sum.of.east.wt(ii) = sum.east.wt
      AW.of.east.unit(ii) = sum.of.east.wt(ii)/ii

''print running average of waiting time and queue length.
  if ii = 1
  print 3 line thus
  ==========  Sum.report routine  ==========
   No    AQn    AWn    AQs    AWs    AQe    AWe
  ==========================================
  always

  print 1 line with ii,
                     AQ.of.north.unit(ii),AW.of.north.unit(ii),
                     AQ.of.south.unit(ii),AW.of.south.unit(ii),
                     AQ.of.east.unit(ii),
                     AW.of.east.unit(ii) thus
  ***   **.**   **.**   **.**   **.**   **.**   **.**
  loop

 release sum.of.north.wt,sum.of.south.wt,sum.of.east.wt
 stop
END ''SUM.REPORT
```

APPENDIX C. A PROGRAM LISTING OF MODEL 2.

```
EVENT NEW.NORTH.UNIT
 define U as an real variable
 create an unit
 let  in = in + 1
 current.time = time.v*hours.v*minutes.v
 arrival.time.of.north.unit(in) = current.time
 U = uniform.f(0,1,s3)

 if north.run <> 0 or n.north.queue > 0
 go to 'case3'
        else
        if U < P1
             next.north.route = 1
             go to 'case1'
        else
             next.north.route = 2
             go to 'case2'

'case1'
 if east.run <> 1
       north.run = 1
       next.north.route = 0
       jn = jn + 1
       schedule an north.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.north.unit(jn) = current.time
       AQ.of.north.unit(jn) = AQ.north.unit
 else
'case3'
       file this unit in the north.queue
       north.queue.length2(in) = n.north.queue
 always
 go to 'end'

'case2'
 if south.run <> 2
       let north.run = 2
       jn = jn + 1
       next.north.route = 0
       schedule an north.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.north.unit(jn) = current.time
       AQ.of.north.unit(jn) = AQ.north.unit
 else
       go to 'case3'
 always

'end'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
```

```
                    next.north.route,next.south.route,
                    next.east.route thus
  NNU  **.**   .**   ** ** **   ** ** **   ** ** **

END   ''NEW.NORTH.UNIT


EVENT NEW.SOUTH.UNIT
 define U as real variable
 create an unit
 is = is + 1
 current.time = time.v*hours.v*minutes.v
 arrival.time.of.south.unit(is) = current.time
 let U = uniform.f(0,1,s3)

 if n.south.queue <> 0 or south.run <> 0
 go to 'case3'
        else
        if U < P1
           next.south.route = 1
           go to 'case1'
        else
           next.south.route = 2
           go to 'case2'

'case1'
 if east.run <> 2
       south.run = 1
       next.south.route = 0
       js = js + 1

       schedule an south.service giving unit
                in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.south.unit(js) = current.time
       AQ.of.south.unit(js) = AQ.south.unit
 else

'case3'
       file this unit in the south.queue
       south.queue.length2(is) = n.south.queue
 always
 go to 'end'

'case2'
 if north.run <> 2
       let south.run = 2
       js = js + 1
       next.south.route = 0
       schedule an south.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.south.unit(js) = current.time
       AQ.of.south.unit(js) = AQ.south.unit
 else
       go to 'case3'
 always
```

```
'end'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
  NSU  **.**   .**   ** ** **   ** ** **   ** ** **

END ''NEW.SOUTH.UNIT


EVENT NEW.EAST.UNIT
 define U as real variable
 create an unit
 ie = ie + 1
 current.time = time.v*hours.v*minutes.v
 arrival.time.of.east.unit(ie) = current.time
 U = uniform.f(0,1,s3)

 if n.east.queue > 0 or east.run <> 0
 go to 'case3'
        else
        if U < P1
             next.east.route = 1
             go to 'case1'
        else
            next.east.route = 2
            go to 'case2'
'case1'
 if north.run <> 1
       east.run = 1
       next.east.route = 0
       je = je + 1
       schedule an east.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.east.unit(je) = current.time
       AQ.of.east.unit(je) = AQ.east.unit
 else
'case3'
       file this unit in the east.queue
       east.queue.length2(ie) = n.east.queue
 always
 go to 'end'

'case2'
 if south.run <> 1
       east.run = 2
       je = je + 1
       next.east.route = 0
       schedule an east.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
       departure.time.of.east.unit(je) = current.time
       AQ.of.east.unit(je) = AQ.east.unit
  else
       go to 'case3'
  always
```

```
'end'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
  NEU  **.**   .**   ** ** **   ** ** **   ** ** **
END ''NEW.EAST.UNIT


EVENT NORTH.SERVICE given unit
 define unit as a pointer variable
 define U as real variable
 destroy this unit
 kn = kn + 1
 current.time = time.v*hours.v*minutes.v
 end.time.of.north.service(kn) = current.time
 U = uniform.f(0,1,s3)

 if n.north.queue > 0 and U < P1
       next.north.route = 1
 always

 if n.north.queue > 0 and U >= P1
       next.north.route = 2
 always

 if north.run = 2
       go to 'service2'
 else
       north.dir(kn) = 1
       north.run = 0

 if next.east.route = 1
       east.run = 1
       je = je + 1
       departure.time.of.east.unit(je) = current.time
       waiting.time.of.east.unit(je) =
          current.time - arrival.time.of.east.unit(je)
       east.queue.length1(je) = n.east.queue
       remove the first unit from the east.queue
       AQ.of.east.unit(je) = AQ.east.unit
       next.east.route = 0
       schedule an east.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always

'next'
 if east.run <> 1 and next.north.route = 1
       jn = jn + 1
       north.run = 1
       departure.time.of.north.unit(jn) = current.time
       waiting.time.of.north.unit(jn)   =
          current.time - arrival.time.of.north.unit(jn)
       north.queue.length1(jn) = n.north.queue
```

```
       remove the first unit from the north.queue
       AQ.of.north.unit(jn) = AQ.north.unit
       next.north.route = 0
       schedule an north.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
always

if next.north.route = 2 and south.run <> 2
       jn = jn + 1
       north.run = 2
       departure.time.of.north.unit(jn) = current.time
       waiting.time.of.north.unit(jn) =
          current.time - arrival.time.of.north.unit(jn)
       north.queue.length1(jn) = n.north.queue
       remove the first unit from the north.queue
       AQ.of.north.unit(jn) = AQ.north.unit
       next.north.route = 0
       schedule an north.service giving unit
                   in gamma.f(alpha2,beta2,s2) minutes
always
go to 'print'

'service2'
north.dir(kn) = 2
north.run     = 0

if next.south.route = 2
       js = js + 1
       south.run = 2
       departure.time.of.south.unit(js) = current.time
       waiting.time.of.south.unit(js) =
          current.time - arrival.time.of.south.unit(js)
       south.queue.length1(js) = n.south.queue
       remove the first unit from the south.queue
       AQ.of.south.unit(js) = AQ.south.unit
       next.south.route = 0
       schedule an south.service giving unit
                in gamma.f(alpha2,beta2,s2) minutes
always
go to 'next'

'print'
print 1 line with current.time,U,
                  north.run,south.run,east.run,
                  n.north.queue,n.south.queue,n.east.queue,
                  next.north.route,next.south.route,
                  next.east.route thus
 NS   **.**   .**   ** ** **   ** ** **   ** ** **
END  ''NORTH.SERVICE


EVENT SOUTH.SERVICE given unit
 define U as real variable
 define unit as a pointer variable
 ks = ks + 1
 destroy this unit
```

```
current.time = time.v*hours.v*minutes.v
end.time.of.south.service(ks) = current.time
U = uniform.f(0,1,s3)

if n.south.queue > 0 and U < P1
      next.south.route = 1
always

if n.south.queue > 0 and U >= P1
      next.south.route = 2
always

if south.run = 2
      go to 'service2'
else
      south.dir(ks) = 1
      south.run     = 0

if next.east.route = 2
      je = je + 1
      east.run = 2
      departure.time.of.east.unit(je) = current.time
      waiting.time.of.east.unit(je) = current.time -
                            arrival.time.of.east.unit(je)
      east.queue.length1(je) = n.east.queue
      remove the first unit from the east.queue
      AQ.of.east.unit(je) = AQ.east.unit
      next.east.route = 0
      schedule an east.service giving unit
               in gamma.f(alpha2,beta2,s2) minutes
always

'next'
if east.run <> 2 and next.south.route = 1
      js = js + 1
      south.run = 1
      departure.time.of.south.unit(js) = current.time
      waiting.time.of.south.unit(js) =
         current.time - arrival.time.of.south.unit(js)
      south.queue.length1(js) = n.south.queue
      remove the first unit from the south.queue
      AQ.of.south.unit(js) = AQ.south.unit
      next.south.route = 0
      schedule an south.service giving unit
                  in gamma.f(alpha2,beta2,s2) minutes
always

if north.run <> 2 and next.south.route = 2
      js = js + 1
      south.run = 2
      departure.time.of.south.unit(js) = current.time
      waiting.time.of.south.unit(js) =
         current.time - arrival.time.of.south.unit(js)
      south.queue.length1(js) = n.south.queue
      remove the first unit from the south.queue
      AQ.of.south.unit(js) = AQ.south.unit
```

```
      next.south.route = 0
      schedule an south.service giving unit
               in gamma.f(alpha2,beta2,s2) minutes
always
go to 'print'

'service2'
south.dir(ks) = 2
south.run     = 0

if next.north.route = 2 and south.run <> 2
      jn = jn + 1
      north.run = 2
      departure.time.of.north.unit(jn) = current.time
      waiting.time.of.north.unit(jn) =
         current.time - arrival.time.of.north.unit(jn)
      north.queue.length1(jn) = n.north.queue
      remove the first unit from the north.queue
      AQ.of.north.unit(jn) = AQ.north.unit
      next.north.route = 0
      schedule an north.service giving unit
                  in gamma.f(alpha2,beta2,s2) minutes
always
go to 'next'

'print'
print 1 line with current.time,U,
                  north.run,south.run,east.run,
                  n.north.queue,n.south.queue,n.east.queue,
                  next.north.route,next.south.route,
                  next.east.route thus
 SS   **.**   .**   ** ** **   ** ** **   ** ** **
END ''SOUTH.SERVICE


EVENT EAST.SERVICE given unit
 define U as real variable
 define unit as a pointer variable
 destroy this unit
 ke = ke + 1
 current.time = time.v*hours.v*minutes.v
 end.time.of.east.service(ke) = current.time
 U = uniform.f(0,1,s3)

 if the east.queue is not empty  and U < P1
       next.east.route = 1
 always

 if the east.queue is not empty and U >= P1
       next.east.route = 2
 always

 if east.run = 2
       go to 'service2'
 else
       east.dir(ke) = 1
```

```
        east.run     = 0

 if next.north.route = 1
        jn = jn + 1
        north.run = 1
        departure.time.of.north.unit(jn) = current.time
        waiting.time.of.north.unit(jn) =
           current.time - arrival.time.of.north.unit(jn)
        north.queue.length1(jn) = n.north.queue
        remove the first unit from the north.queue
        AQ.of.north.unit(jn) = AQ.north.unit
        next.north.route = 0
        schedule an north.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always

'next'
 if north.run <> 1 and next.east.route = 1
        je = je + 1
        east.run =1
        departure.time.of.east.unit(je) = current.time
        waiting.time.of.east.unit(je) =
           current.time - arrival.time.of.east.unit(je)
        east.queue.length1(je) = n.east.queue
        remove the first unit from the east.queue
        AQ.of.east.unit(je) = AQ.east.unit
        next.east.route = 0
        schedule an east.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always

 if south.run <> 1 and next.east.route = 2
        je = je + 1
        east.run = 2
        departure.time.of.east.unit(je) = current.time
        waiting.time.of.east.unit(je) =
           current.time - arrival.time.of.east.unit(je)
        east.queue.length1(je) = n.east.queue
        remove the first unit from the east.queue
        AQ.of.east.unit(je) = AQ.east.unit
        next.east.route = 0
        schedule an east.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always
 go to 'print'

'service2'
 east.dir(ke) = 2
 east.run = 0

 if next.south.route = 1
        js = js + 1
        south.run = 1
        departure.time.of.south.unit(js) = current.time
        waiting.time.of.south.unit(js) =
           current.time - arrival.time.of.south.unit(js)
```

```
        south.queue.length1(js) = n.south.queue
        remove the first unit from the south.queue
        AQ.of.south.unit(js) = AQ.south.unit
        next.south.route = 0
        schedule an south.service giving unit
                    in gamma.f(alpha2,beta2,s2) minutes
 always
 go to 'next'

'print'
 print 1 line with current.time,U,
                   north.run,south.run,east.run,
                   n.north.queue,n.south.queue,n.east.queue,
                   next.north.route,next.south.route,
                   next.east.route thus
  ES   **.**   .**   ** ** **   ** ** **   ** ** **

END ''EAST.SERVICE
```

LIST OF REFERENCES

1. Kleinrock, L., *Queueing Systems Vol. I and II*, John Wiely & Sons, Inc., 1975.

2. Fishman, G. S., *Principles of Discrete Event Simulation*, John Wiley & Sons, Inc., 1978.

3. Taylor, H. M. and S. Karlin, *An Introduction to Stochastic Modeling*, Academic Press, Inc., 1984.

4. Ross, S. M., *Introduction to Probability Models*, Academic Press, Inc., 1985.

5. Olkin, I., L. J. Gleser and C. Derman, *Probability Models and Applications*, Macmillan Publishing Co. Inc., 1980.

6. Lewis, P. A. W. and E. J. Orav., *Simulation Methodology for Statisticians, Engineers and Operations Analysists Vol.I*, 1987.

7. Braun, J. (editor), *The SIMSCRIPT II.5 Reference Handbook, Second Edition*, CACI, 1983.

8. Mullarney, A. (editor), P. J. Kiviat, R. Villanueva and H. M. Markowitz, *The SIMSCRIPT II.5 Programming Language*, CACI, 1983.

9. Law, A. M. and C. S. Larmey, *An Introduction to Simulation Using SIMSCRIPT II.5*, CACI, 1984.

10. Russel, E. C., *Building Simulation Models with SIMSCRIPT II.5*, CACI, 1983.

11. Lavenberg, S. S., *Computer Performance Modeling Handbook*, Academic Press, Inc., 1983.

12. Hartman, J. K., *Lecture Notes in Aggregated Combat Modelling*, Naval Postgraduate School, Monterey, CA, 1985.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Professor Peter Purdue, Chairman, Code 55Pd Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
4.	Professor Peter A. W. Lewis, Code 55Lw Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
5.	Professor Keebom Kang, Code 55Ka Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
6.	Jim Stevens, Code 55 Department of Operations Research Naval Postgraduate School Monterey, CA 93943-5000	1
7.	Library, P.O.Box 77 K.M.A, Gong Neung Dong, Dobong Ku Seoul, 130-09 Republic of Korea	1
8.	Library, P.O.Box 2 R.O.K.A HQs. Yong San Dong, Yong San Gu Seoul, 140-01 Republic of Korea	1
9.	System Analysis Department in R.O.K.A HQs. P.O.BOX 2 Yong San Dong, Yong San Gu Seoul, 140-01 Republic of Korea	1

10. Park, Hun Keun 1
SMC #2997
Naval Postgraduate School
Monterey, CA 93943

11. Lee, Hyeon Soo 1
SMC #2337
Naval Postgraduate School
Monterey, CA 93943

12. Choi, Mun Soo 1
SMC #1840
Naval Postgraduate School
Monterey, CA 93943

13. Lee, Jae Yeong 1
Simgok-1-Dong 585, 10/4
Bucheon City, Nam-gu
Kyunggi-Do, 422-011
Republic of Korea

14. Park, Byung Goo 1
Postal Code 130-12
229 Dae Ssang Ryung-Ri,
Cho Wol-Myun, Kwang Ju-Kun, Kyung Ki-Do
Republic of Korea

15. Lee, Chong Ho 1
Nammun-1-Ri, 3-Ban
Yangyang-Eup, Yangyang-Gun
Kangwon-Do,
Republic of Korea

16. Song, Chong Chul 2
Won Sang Dae Burak, Sang Dae Ri,
Po Du Moeyn, Ko Heong Kun, Cheon Nam, 548-880
Republic of Korea